Harald Baumgart

Höhere Mathematik auf dem CPC 464



Harald Baumgart

Höhere Mathematik auf dem CPC 464

CHIP WISSEN

Harald Baumgart

Höhere Mathematik auf dem CPC 464

Programme mit Erklärungen des Lösungsweges und Beispielen



VOGEL-BUCHVERLAG
WÜRZBURG

Dipl.-Ing. Harald Baumgart
geboren 1957 in Idsingen. Studium der Verfahrenstechnik in Hamburg. Beschäftigt in der Forschung und Entwicklung; hier u.a. Erstellung und Mitarbeit an Programmen zur Versuchsdurchführung und -auswertung.

CIP-Kurztitelaufnahme der Deutschen Bibliothek

Baumgart, Harald:
Höhere Mathematik auf dem CPC 464: Programme mit Erklärungen d. Lösungsweges u. Beispielen/ Harald Baumgart. – 1. Aufl. – Würzburg: Vogel, 1985.
(Chip-Wissen)
ISBN 3-8023-0856-5

ISBN 3-8023-0856-5
1. Auflage. 1985

Printed in Germany

Umschlaggestaltung: Bernd Schröder, Böhl
Satz: Schmitt u. Köhler, Würzburg
Druck: Alois Erdl KG, Trostberg

Inhaltsverzeichnis

Vorwort

Dieses Buch entstand aus einer Sammlung von Programmen, die geschrieben wurden, um den vorhandenen Computer zu nützlicheren Dingen zu überreden als Strom-, Gas- und Wasserabrechnungen bzw. Gewichtsverfolgungen der neugeborenen Tochter. Was lag also näher als zu versuchen, den Rechner für Probleme einzusetzen, mit denen man sich ansonsten lang und breit (und mit einer gehörigen Portion Frust) «zu Fuß» auseinandersetzen mußte; so entstanden Programme zur Ausgleichsrechnung, zur Fehleranalyse und zur Funktionsbetrachtung. Da das Schreiben dieser Programme gerade für den CPC 464 sehr viel Spaß machte und dabei immer wieder neue (gute!) Seiten dieses Computers festgestellt werden konnten, wurde die Programmsammlung ergänzt um Problemkreise, die laut Lehrplan bzw. aus eigener Erfahrung in der Oberstufe der Gymnasien, in den Fachoberschulen und in den ersten Semestern eines Ingenieurstudiums anliegen.

Demzufolge ist dieses Buch gedacht für alle Schüler ab 11. Klasse, für alle Studenten, die sich im Grundstudium mit Mathematik herumschlagen müssen, aber auch für alle «fertigen» Techniker und Ingenieure. Es soll aber keine reine Programmsammlung darstellen; vielmehr werden die mathematischen Grundlagen zu jedem Programm möglichst eindeutig erläutert, damit man auch weiß wieso, warum, weshalb. Hierzu gehört die Durchrechnung von Beispielen ohne Benutzung des Rechners. Vorausgesetzt werden in etwa Mathematikkenntnisse der 10. Klasse sowie – in einigen Kapiteln – Grundkenntnisse der Differentialrechnung. Des weiteren werden vielfach Literaturangaben gemacht, die mit einem G für mehr grundlagenorientierte oder mit einem W für weiterführende Literatur versehen sind.

Der Schwerpunkt des Buches liegt aber ganz eindeutig bei den Programmen selbst; diese können selbstverständlich auch ohne vorheriges Durcharbeiten der Grundlagenabschnitte benutzt werden. Außerdem

wurden viele REM-Anweisungen eingebaut und auf gute Dialogtechnik geachtet (bei dem Speicherplatz des CPC 464 ist das ja kein Problem); die Bedienung der Programme ist also recht einfach, und eventuelle Änderungen sollten kein Problem sein. Diesem Zweck dienen auch die beigestellten Programmbeschreibungen.

Möge dieses Buch dazu beitragen, das Grauen von der Mathematik ein wenig abzubauen. Und wie sagte schon C. F. GAUSS:

Der Mangel an mathematischer Bildung gibt sich durch nichts so auffallend zu erkennen wie durch maßlose Schärfe im Zahlenrechnen.

Daher nicht erst auf die 16-bit-Computergeneration warten (die sind sowieso viel zu teuer), sondern ran mit den 8 bits von Schneider!

Appen

Harald Baumgart

1
Funktionen

1.1 Darstellung beliebiger Funktionen

Das Ziel einer jeden Kurvendiskussion ist es, einen prinzipiellen Überblick über den Verlauf einer Funktion zu erhalten. Üblicherweise wird so vorgegangen, daß besondere Punkte wie Nullstellen, Extremwerte, Pole und Asymptoten berechnet werden und als Abschluß der Graph der Funktion gezeichnet wird.

Mit Hilfe eines Computers kann man diese Kurvendiskussion auch umgekehrt durchführen. Mathematiklehrer sind darüber zwar nicht sonderlich erbaut, aber es erleichtert die Arbeit ungemein, wenn man sich als *ersten* Schritt den Funktionsgraph erzeugt. Das hat zum einen den Vorteil, daß man einfach weiß, wie die Kurve verläuft, zum anderen kann man für Nullstellen usw. zumindest Schätzwerte entnehmen und so die nachfolgenden Rechnungen überprüfen. Für einen Computer spielt die Anzahl der Rechenschritte nur eine untergeordnete Rolle. Demzufolge geht das Programm so vor, daß die Funktionswerte in sehr kleinen Intervallen berechnet und auf dem Bildschirm dargestellt werden. Hierbei kann der CPC 464 seine hervorragenden grafischen Fähigkeiten voll ausspielen.

In den Zeilen

Zeilen	10	
bis	270	ist das eigentliche Hauptprogramm abgelegt. Hierbei dienen
Zeile	20	
und	30	dazu, den Rechner zu veranlassen, bei einem beliebigen Fehler, der bei der Rechnung auftritt, nicht mit einer Fehlermeldung auszusteigen, sondern den Fehler zu ignorieren und weiterzumachen. Fehler bei Berechnungen können des öfteren auftreten, da Funktionen durchaus

nicht im gesamten Wertebereich definiert sein müssen. Erinnert sei hier an Pole oder Lücken von gebrochen rationalen Funktionen, aber auch an die Wurzelfunktion, die innerhalb der Menge der reellen Zahlen nicht definiert ist, wenn der Wert unter der Wurzel in den negativen Bereich übergeht. In

Zeile 60 ist die Funktion abgelegt. Diese Zeile wird als Unterprogramm aufgerufen, wenn ein Funktionswert berechnet werden soll.

Da der CPC 464 Schwierigkeiten beim Potenzieren negativer Zahlen hat, sollte man z.B. x^3 nicht als x^3, sondern als $x*x*x$ eingeben. Auf dieses Problem wird später noch eingegangen.

Wenn eine Funktion auf dem Bildschirm dargestellt werden soll, sind bestimmte Grunddaten für das Achsenkreuz wie Darstellungsbereich bezüglich x- und y-Achse, Beschriftungsabstand usw. nötig. In den

Zeilen 80
bis 110 werden diese Daten per DATA-Anweisung eingegeben. Daneben besteht auch die Möglichkeit, diese Daten im Dialog einzugeben; die Abfrage, welche Art der Dateneingabe gewählt werden soll, liegt in den

Zeilen 140
bis 180 Sollen die Daten per DATA-Anweisung eingelesen werden, so erfolgt zunächst der Sprung ins Unterprogramm in den

Zeilen 280
bis 310 Bevor das eigentliche Lesen durchgeführt wird, muß zunächst noch der Zeiger für die READ-Anweisung durch

Zeile 290 auf Zeile 100 gesetzt werden, damit auch bei mehrmaligem Durchlauf die richtigen Daten verwendet werden.

Sollen die Daten dagegen im Dialog eingegeben werden, springt das Programm ins Unterprogramm

Zeile 320
bis 530 Aus diesen Zeilen kann man auch entnehmen, welche Bedeutung die einzelnen Größen haben, da in der READ/DATA-Anweisung bzw. im Dialog natürlich die gleichen Variablennamen verwendet werden. An dieser Stelle sei außerdem noch darauf hingewiesen, daß an den großen

Strichen der x- und y-Achse, also dort, wo beide Achsen *grob* geteilt werden (*dxg* und *dyg*), auch eine Beschriftung durch einen Zahlenwert erfolgt; die Skalierung wird hierbei automatisch durchgeführt. In den

Zeilen 540
bis 770 besteht zunächst die Möglichkeit, sich für den eingegebenen Wertebereich der x-Achse den maximalen und den minimalen y-Wert berechnen zu lassen. Wenn der Rechner hiermit fertig ist, werden in

Zeile 700
bis 710 beide Werte sowie die entsprechenden Werte für die y-Achse ausgegeben, die man ja vorher entweder über READ/DATA oder im Dialog dem CPC 464 mitgeteilt hat. In

Zeile 720
bis 760 besteht jetzt die Möglichkeit, die Werte für die y-Achse zu ändern; hat man sie im Vorwege zu groß oder zu klein gewählt, kann man sie hier noch anpassen, damit die Kurve auf dem Bildschirm optimal dargestellt wird. In den

Zeilen 780
bis 800 erfolgt die Festlegung des Koordinatenursprungs auf dem Bildschirm. Hier wird versucht, diesen so gut wie möglich auszunutzen. In den

Zeilen 870
bis 920 wird zunächst das Koordinatenkreuz gezeichnet und nachfolgend die Achsen eingeteilt und beschriftet. Im Unterprogramm

Zeile 1490
bis 1610 werden schließlich die Funktionswerte berechnet und nach Anpassung an das Bildschirmformat graphisch ausgegeben. Mit Hilfe des letzten Unterprogramms in

Zeile 1620
bis 1800 besteht noch die Möglichkeit, Text in das Diagramm einzugeben; dies kann z.B. irgendeine Überschrift, die Funktionsbezeichnung, das Datum o.ä. sein. Durch Cursorbewegung besteht noch die Möglichkeit, Text in das Diagramm einzugeben; dies kann z.B. irgendeine Überschrift, die Funktionsbezeichnung, das Datum o.ä. sein. Durch Cursorbewegung wird der Text an eine beliebige

Stelle des Bildschirms gebracht und dort ausgedruckt. Der Vorgang kann beliebig oft wiederholt werden; auf diese Weise kann man auch z.B. Achsenbezeichnungen löschen oder ändern.

Den Abschluß des Programms bildet

Zeile 270 Durch sie verschwindet der Cursor. Beendet wird das Programm durch zweimaliges Drücken der ESC-Taste.

Ohne auf jede einzelne Zeile hinzuweisen, sollen an dieser Stelle noch einige Standard-Programmierweisen besprochen werden, wie sie auch in allen anderen Programmen dieses Buches vorkommen.

Die erste sei die Eingabe einer Codezahl, aufgrund der das Programm Entscheidungen trifft und z.B. in bestimmte Unterprogramme verzweigt. Im Programm "Funktion 1.0" kommt das in den

Zeilen 150
bis 180 vor; sehr viel ausgeprägter ist es in den Menüprogrammen ab Kapitel 4. Die Codezahl wird mit einer Kurzbeschreibung ihrer Funktion ausgegeben, diese Aufgabe erfüllen

Zeile 150
und 160

Wegen der INPUT-Anweisung in

Zeile 170 wartet das Programm dann auf die Eingabe der Codezahl; dies erfolgt hier über die Variable *a*.

Gleich nach Eingabe der Codezahl wird geprüft, ob sich *a* innerhalb des vorgegebenen Wertebereiches befindet, da ansonsten eine Fehleingabe vorliegt. In diesem Fall wird der gesamte Text erneut ausgegeben, so daß eine neue Eingabe erfolgen muß. Wird statt einer Zahl z.B. ein Buchstabe oder ein sonstiges Zeichen eingetippt, erkennt der Rechner diesen Fehler von sich aus und verlangt die Eingabe einer Zahl.

Wenn diese Prüfung durchgeführt worden ist, kann

Zeile 180 in die entsprechenden Unterprogramme verzweigen.

Die zweite besprochene Standardprogrammierweise sei eine "Ja/Nein-Abfrage"; diese kommt z.B. in den

Zeilen 1750
bis 1780 vor.

Auch hier wird zunächst ein Text ausgegeben, aufgrund

dessen die Ja/Nein-Entscheidung ausgeführt wird. Das Programm erwartet die Eingabe von *a*$, also einer Stringvariablen. In diesem Buch wird generell die Ja-Anwort durch Betätigung einer der beiden ENTER-Tasten gegeben. Dies hat den Grund, daß zumindest die eine ENTER-Taste beim CPC 464 erfreulich groß ausgefallen ist, so daß eine Betätigung besonders komfortabel ist. Im allgemeinen kommt die Ja-Antwort wesentlich häufiger vor als die Nein-Antwort; diese gibt man durch Betätigen der n-Taste.
Falls die ENTER-Taste gedrückt wurde, springt

Zeile 1760 zur entsprechenden Programmstelle; das gleiche macht

Zeile 1770 für die n-Taste. Wichtig ist hier, daß beide möglichen Antworten getrennt abgefragt werden. Wurde nämlich versehentlich ein anderes Zeichen eingegeben, gelangt das Programm wegen der vorhergehenden Abfragen nach

Zeile 1780 und die Eingabe wird nicht akzeptiert, sondern der Abfragetext wird erneut ausgegeben. Fehleingaben sind somit ausgeschlossen.

Um die Arbeitsweise des Programms zu zeigen, ist in Zeile 60 die Funktion

$$y = 3x^3 - 10x^2 + 3$$

abgelegt. Wählt man die Darstellung über die DATA-Anweisung, so wird die Funktion im Wertebereich -2 bis $+4$ wiedergegeben. Die y-Achse geht von -15 bis $+5$.

Länge des Programms: etwa 6,7 K.

```
10 REM Programm Funktion 1.0
20 ON ERROR GOTO 30
30 RESUME NEXT
40 GOTO 120
50 REM Zeile 60 = reserviert fuer die Funktionsgleic
hung
60 y = 3*x*x*x-10*x*x+3
70 RETURN
80 REM 100=DATA fuer die Grunddaten Koordinatenkreuz

90 REM Ablegeweise: DATA xmin,xmax,ymin,ymax,dxg,dxk
,dyg,dyk,darstx,darsty
100 DATA -2,4,-15,5,6,2,4,5,1,3
110 REM Moegliche Werte fuer darstx bzw. darsty: 0 =
 keine Darstellung ; 1 = 1Dezimalstelle ; 2 = 2 Dezi
malstellen ; 3 = keine Dezim
alstelle
120 MODE 2
130 PRINT"Programm Funktion 1.0 zur Darstellung eine
r beliebigen Funktion ":PRINT"in einem Koordinatenkr
euz"
140 LOCATE 1,10 :PRINT"Liegen die Grunddaten fuer da
s Achsenkreuz (xmin,xmax etc.) dem Programm in     e
inerDATA-Anweisung vor oder
sollen sie im Dialog eingegeben werden ?"
150 PRINT: PRINT TAB(20);"DATA";TAB(50);"(1)"
160 PRINT:PRINT TAB(20);"Dialog";TAB(50);"(2)"
170 INPUT a : IF a<1 OR a>2 THEN 140
180 ON a GOSUB 280,320
190 GOSUB 540 : REM Berechnung und Ausgabe von ymin
und ymax
```

```
200 GOSUB 780 :REM Berechnung der Bildschirmpunkte f
uer den Koordinatenursprung
210 GOSUB 870 : REM Zeichnung des Koordinatenkreuzes
220 GOSUB 930 : REM Einteilung in dxg,dyg
230 GOSUB 1030 : REM Einteilung in dxk,dyk
240 GOSUB 1130 : REM Beschriftung der Achsen
250 GOSUB 1490 : REM Berechnung und Ausgabe der Funk
tionswerte
260 GOSUB 1620 : REM Texteintragung ins Diagramm
270 GOTO 270
280 REM UP Lesen der Grunddaten Koordinatensystem pe
r DATA
290 RESTORE 100
300 READ xmin,xmax,ymin,ymax,dxg,dxk,dyg,dyk,darstx,
darsty
310 RETURN
320 REM UP Eingabe der Grunddaten Koordinatensystem
im dialog
330 CLS : PRINT"Min-Wert der X-Achse : ";:INPUT xmin
340 PRINT:PRINT"Max-Wert der X-Achse : ";:INPUT xmax
350 PRINT:PRINT"Min-Wert der Y-Achse : ";:INPUT ymin

360 PRINT:PRINT"Max-Wert der Y-Achse : ";:INPUT ymax
370 PRINT:PRINT:PRINT:PRINT"Wie oft soll die X-Achse
  g r o b  (mit Beschriftung) geteilt werden : ";:IN
PUT dxg
380 PRINT:PRINT"Wie oft soll ein Feld noch  f e i n
 (ohne Beschriftung) unterteilt werden : ":INPUT dxk

390 PRINT:PRINT"Wie oft soll die Y-Achse  g r o b  (
mit Beschriftung) geteilt werden : ";:INPUT dyg
```

```
400 PRINT:PRINT"Wie oft soll ein Feld noch  f e i n
 (ohne Beschriftung) unterteilt werden : ":INPUT dyk

410 CLS:PRINT"Wie sollen die Zahlen an der x-Achse d
argestellt werden ?":
420 PRINT:PRINT:PRINT TAB(20);"-gar keine Darstellun
g";TAB(60);"(0)"
430 PRINT:PRINT TAB(20);"-mit einer Dezimalstelle";T
AB(60);"(1)"
440 PRINT:PRINT TAB(20);"-mit zwei Dezimalstellen";T
AB(60);"(2)"
450 PRINT:PRINT TAB(20);"-ohne Dezimalstellen";TAB(6
0);"(3)"
460 INPUT darstx : IF darstx<0 OR darstx>3 THEN 410
470 CLS:PRINT"Wie sollen die Zahlen an der y-Achse d
argestellt werden ?"
480 PRINT:PRINT:PRINT TAB(20);"-gar keine Darstellun
g";TAB(60);"(0)"
490 PRINT:PRINT TAB(20);"-mit einer Dezimalstelle";T
AB(60);"(1)"
500 PRINT:PRINT TAB(20);"-mit zwei Dezimalstellen";T
AB(60);"(2)"
510 PRINT:PRINT TAB(20);"-ohne Dezimalstellen";TAB(6
0);"(3)"
520 INPUT darsty : IF darsty<0 OR darsty>3 THEN 470
530 RETURN
540 REM UP Berechnung und Ausgabe von ymin und ymax
550 CLS : PRINT"Sollen ymin und ymax berechnet werde
n ( /n) ";:INPUT a$
560 IF a$="" THEN 590
570 IF a$="n" THEN 770
```

```
580 GOTO 540
590 a=-1
600 PRINT:PRINT:PRINT"Bitte einen Augenblick Geduld.
 Ich arbeite."
610 PRINT:PRINT"Wenn ich bei 500 bin, bin ich fertig
."
620 yminb=0 : ymaxb=0
630 FOR x=xmin TO xmax STEP (xmax-xmin)/500
640 a=a+1
650 LOCATE 1,23 : PRINT a
660 GOSUB 60
670 IF y<yminb THEN yminb=y
680 IF y>ymaxb THEN ymaxb=y
690 NEXT x
700 CLS : PRINT"Bislang im Programm verwendete Werte
 fuer die Einteilung der y-Achse : ":PRINT:PRINT"ymi
n = ";ymin,"ymax = ";ymax
710 PRINT:PRINT:PRINT"Soeben aus der Funktionsgleich
ung berechnete Werte :":PRINT:PRINT"ymin = ";yminb,"
ymax = ";ymaxb
720 PRINT:PRINT:PRINT"Sollen die Werte der Y-Achsene
inteilung veraendert werden ( /n) ";:INPUT a$
730 IF a$="" THEN 760
740 IF a$="n"THEN 770
750 GOTO 700
760 PRINT:PRINT"ymin = ";:INPUT ymin :PRINT:PRINT"ym
ax = ";:INPUT ymax
770 RETURN
780 REM UP Berechnung der Bildschirmpunkte fuer den
Koordinatenursprung
790 REM Erlauterungen fuer die Umrechnung auf Bildsc
```

```
hirm s. UP Funktionsdarstellung
800 xb=  70+(500/(xmax-xmin))*(0-xmin)
810 IF xmin>=0 THEN xb = 70
820 IF xmax<=0 THEN xb = 570
830 yb = 40+(320/(ymax-ymin))*(0-ymin)
840 IF ymin >= 0 THEN yb = 40
850 IF ymax <= 0 THEN yb = 360
860 RETURN
870 REM UP Zeichnen des Koordinatenkreuzes
880 REM Erlauterungen fuer die Umrechnung auf Bildsc
hirm s. UP Funktionsdarstellung
890 CLS
900 ORIGIN 60,yb : DRAWR 520,0
910 ORIGIN xb,30 : DRAWR 0,340
920 RETURN
930 REM UP Einteilung in dxg,dyg
940 FOR k=0 TO dxg
950 ORIGIN 70+k*(500/dxg),yb-10
960 DRAWR 0,20
970 NEXT k
980 FOR k=0 TO dyg
990 ORIGIN xb-10,40+k*(320/dyg)
1000 DRAWR 20,0
1010 NEXT k
1020 RETURN
1030 REM UP Einteilung in dxk,dyk
1040 FOR k=0 TO dxg*dxk
1050 ORIGIN 70+k*(500/(dxg*dxk)),yb-5
1060 DRAWR 0,10
1070 NEXT k
1080 FOR k=0 TO dyg*dyk
```

```
1090 ORIGIN xb-5,40+k*(320/(dyg*dyk))
1100 DRAWR 10,0
1110 NEXT k
1120 RETURN
1130 REM UP Beschriftung der ACHSEN
1140 TAG
1150 REM Beschriftung y-Achse
1160 IF xb = 570 THEN 1240
1170 FOR k=0 TO dyg
1180 ORIGIN xb-70,46+k*(320/dyg),xb-70,xb-15,400,0
1190 f=ymin+k*((ymax-ymin)/dyg)
1200 IF f=0 AND xb<>70 THEN 1220
1210 ON darsty GOSUB 1460,1470,1480
1220 NEXT k
1230 GOTO 1290
1240 FOR k=0 TO dyg
1250 ORIGIN 585,46+k*(320/dyg),585,640,400,0
1260 f=ymin+k*((ymax-ymin)/dyg)
1270 ON darsty GOSUB 1460,1470,1480
1280 NEXT k
1290 REM Beschriftung x-Achse
1300 IF yb = 360 THEN 1380
1310 FOR k=0 TO dxg
1320 ORIGIN 27+k*(500/dxg),yb-15,27+k*(500/dxg),87+k
*(500/dxg),yb-15,yb-40
1330 f=xmin+k*((xmax-xmin)/dxg)
1340 IF f=0 AND yb<>360 THEN 1360
1350 ON darstx GOSUB 1460,1470,1480
1360 NEXT k
1370 GOTO 1430
1380 FOR k=0 TO dxg
```

```
1390 ORIGIN 27+k*(500/dxg),390,27+k*(500/dxg),87+k*(
500/dxg),395,375
1400 f=xmin+k*((xmax-xmin)/dxg)
1410 ON darstx GOSUB 1460,1470,1480
1420 NEXT k
1430 TAGOFF
1440 ORIGIN 0,0,0,640,400,0
1450 RETURN
1460 PRINT USING "#####.#";f;:RETURN
1470 PRINT USING "####.##";f;:RETURN
1480 PRINT USING "#######";f;:RETURN
1490 REM UP Berechnung und Ausgabe der Funktionswert
e
1500 REM Funktionsdarstellung im "window" 70,570,360
,40
1510 REM Achsenkreuze werden jeweils um 10 Punkte la
enger ausgezogen
1520 REM delta x = 500, delta y = 320
1530 ORIGIN 70,40
1540 a=0 : y=0 : IF ymin>=0 THEN y=ymin
1550 FOR x=xmin TO xmax STEP (xmax-xmin)/500
1560 a = a+1
1570 GOSUB 60 : REM Berechnung des Funktionswertes
1580 b = 320/(ymax-ymin)*(y-ymin)
1590 PLOT a,b
1600 NEXT x
1610 RETURN
1620 REM UP Texteintragung ins Diagramm
1630 IF yb=40 THEN WINDOW#1,1,80,1,1 : GOTO 1650
1640 WINDOW#1,1,80,25,25
1650 PRINT#1,"Soll in das Diagramm Text eingetragen
```

```
werden ( /n) ";:INPUT#1,a$
1660 IF a$="" THEN 1690
1670 IF a$="n" THEN 1790
1680 GOTO 1650
1690 CLS #1
1700 PRINT#1,"Text";:INPUT#1,a$
1710 PRINT#1,"Wohin soll der Text? (Cursor bewegen,
dann ENTER-Taste druecken)"
1720 LOCATE 1,1
1730 INPUT;b$ : IF b$="" THEN PRINT a$ : GOTO 1750
1740 GOTO 1710
1750 LOCATE 1,1 : PRINT" ": CLS#1 : PRINT#1,"Noch ei
n Text ( /n) ";:INPUT#1,a$
1760 IF a$="" THEN 1690
1770 IF a$="n" THEN 1790
1780 GOTO 1750
1790 CLS #1
1800 RETURN
```

1.2 Berechnung einer Wertetabelle

Neben der Darstellung des Funktionsgraphen ist auch eine Wertetabelle ein nützliches Hilfsmittel, um einen Überblick über den prinzipiellen Verlauf einer Funktion zu bekommen. Außerdem können für besonders interessierende x-Werte die dazugehörigen Funktionswerte entnommen werden, so daß inter- und extrapoliert werden kann und die genauen Werte für Extremwerte, Wendestellen usw. festgehalten werden können.

Das Programm *Wertetab 1.0* benutzt wie fast alle anderen Programme in diesem Buch den Mode 2, d.h., es werden 80 Zeichen pro Zeile auf dem Bildschirm dargestellt.

Die Funktionsgleichung ist in

Zeile 70 abgelegt; genau wie im Programm *Funktion 1.0* wird sie als Unterprogramm in das übrige Programm eingebunden. In den

Zeilen 90	
bis 140	wird der Bildschirm in fünf Teilbereiche eingeteilt, wobei die ersten drei für die Ausgabe der Wertetabelle benutzt werden, der vierte für die Eingabe von x-Werten und der fünfte für die Ausgabe von Text zwecks Bedienerführung. Durch die Benutzung dieser WINDOWs bietet das Programm die Möglichkeit, bis zu drei formatierte Wertetabellen auf dem Bildschirm auszugeben, wobei bis zu 20 Werte in eine Tabelle eingetragen werden können. Daneben können aber auch die berechneten Funktionswerte ohne Formatierung auf dem Bildschirm auszugeben. Die Abfrage hierfür findet in den
Zeilen 150	
bis 200	statt. Für die formatierte Ausgabe muß man über die
Zeilen 210	
bis 290	angeben, wie die x-Werte in der Tabelle dargestellt werden sollen; die entsprechende Codezahl wird als *darstx* dem Programm eingegeben. In den
Zeilen 300	
bis 380	erfolgt das Entsprechende für die y-Werte; die Codezahl ist hier *darsty*. In den
Zeilen 420	
bis 480	wird der Kopf der Wertetabelle gezeichnet; über die FOR/NEXT-Schleife in den
Zeilen 410	
und 590	bzw. über die Berechnung der x-Koordinaten in den
Zeilen 420	
bis 450	wird der gleiche Programmabschnitt für alle drei Wertetabellen benutzt, obwohl sie an verschiedenen Plätzen auf dem Bildschirm erscheinen. In
Zeile 500	wird schließlich der x-Wert eingegeben, für den ein Funktionswert berechnet werden soll; nach Überprüfung der Endbedingung (s.u.) in
Zeile 520	geschieht dies in
Zeile 540	Die Ausgabe erfolgt in den
Zeilen 550	

bis 570 wobei entsprechend der Codezahlen *darstx* und *darsty* die Formatierung gewählt wird, die in den Unterprogrammen in den

Zeilen 610

bis 720 abgelegt ist. Für *x* wird hierbei wegen der Rechtsbündigkeit PRINT USING benutzt, während für *y* ROUND Verwendung findet und die Zahlen somit linksbündig ausgegeben werden.

Sollen die Funktionswerte unformatiert ausgegeben werden, müssen die Programmschritte ab

Zeile 700 benutzt werden. Hierbei wird in

Zeile 770 ein gesondertes Ausgabefenster definiert; in

Zeile 790 erfolgt die Eingabe des *x*-Wertes und in

Zeile 800 die Berechnung des Funktionswertes.

Erfolgt bei der Berechnung des Funktionswertes ein Fehler (z.B. Division durch 0), so wird über

Zeile 20

und 30 der Funktionswert auf 999999999 gesetzt und normal weitergemacht.

Programmlänge: etwa 2.8 K.

Anhand des Programms *Wertetab 1.0* sei beispielhaft das Beenden der Eingabe einer Anzahl von Meßwerten erläutert. Für diese Aufgabe gibt es diverse Lösungsmöglichkeiten. Am einfachsten ist es, wenn die Anzahl der einzugebenden Meßwerte bekannt ist; in diesem Fall baut man entweder einen Zähler auf oder benutzt gleich eine FOR/NEXT-Schleife. Wenn die Anzahl der Meßwerte bei mehreren Programmabläufen nicht konstant ist, muß man vor der eigentlichen Meßwerteingabe die für den jeweiligen Durchlauf gültige Zahl als Variable eingeben und diese Variable im Zähler benutzen.

Eine weitere Möglichkeit zum Beenden der Eingabe besteht darin, daß man im Programm einen Abbruchwert definiert. Dies kann eine beliebige Zahl sein; allerdings sollte sie so gewählt werden, daß sie mit Sicherheit nicht als Meßwert vorkommt. Wenn man jetzt einen Meßwert eingegeben hat, prüft das Programm zunächst, ob der Eingabewert gleich dem Abbruchwert ist. Ist das der Fall, hört das Programm

mit der Meßwerteingabe auf und führt mit der Weiterverarbeitung o.ä. fort.
Das soeben beschriebene Verfahren wird z.B. im Programm *Ausgleich* (siehe Kapitel 10) verwendet. Der Abbruchwert ist dort 333.33 (Zeile 230). Hier im Programm *Wertetab 1.0* wird eine Variante des Verfahrens angewendet. Als Abbruchwert wird keine Zahl, sondern der Buchstabe *n* verwendet. Das hat den Vorteil, daß der Abbruchwert garantiert nicht als Eingabewert vorkommt; man muß allerdings in Kauf nehmen, daß die Meßwerteingabe etwas komplizierter wird. Da der Abbruchwert ein String ist, muß auch der jeweilige Meßwert zunächst als Stringvariable eingegeben werden, da sonst kein Vergleich stattfinden kann. Dieser wird in

Zeile 520 durchgeführt; wurde *n* eingegeben, beendet das Programm die Eingabe von Werten.
Ist die Abbruchbedingung nicht gegeben, so wird der String *x*$ zunächst in die Zahl *x* umgewandelt; dies passiert in

Zeile 530 unter Benutzung der VAL-Funktion. Anschließend macht das Programm dann mit Berechnung des Funktionswertes, Ausgabe usw. weiter.

```
10 REM Programm Wertetab 1.0
20 ON ERROR GOTO 30
30 y=999999999:RESUME NEXT
40 MODE 2
50 GOTO 90
60 REM Zeile 60 = Funktionsgleichung
70 y=SQR(x)
80 RETURN
90 REM Bildschirm wird in 5 Bereiche eingeteilt
100 WINDOW#1, 1,27,1,22
110 WINDOW #2, 28,54,1,22
120 WINDOW #3,55,80,1,22
```

```
130 WINDOW #4,1,80,23,24 : REM fuer die Eingabe der
werte etc.
140 WINDOW#5,1,80,25,25 : REM Fuer Ausgabe von Text
150 CLS
160 PRINT"Programm Wertetabelle"
170 PRINT:PRINT:PRINT:PRINT"Sollen die Werte in eine
r  formatierte Tabelle ausgegeben werden           (
1) ":PRINT"oder sollen die Y
-Werte jeweils einzeln ausgegeben werden
      (2)":INPUT a
180 IF a<1 OR a>2 THEN 150
190 IF a=1 THEN 210
200 IF a=2 THEN 760
210 CLS
220 PRINT"Wie soll der X-Wert in der Tabelle darsges
tellt werden ?":PRINT:PRINT
230 PRINT TAB(20);"-mit einer Dezimalstelle   ";TAB
(60);"(1)"
240 PRINT TAB(20);"-mit zwei Dezimalstellen ";TAB (6
0);"(2)"
250 PRINT TAB(20);"-mit drei Dezimalstellen ";TAB (6
0);"(3)"
260 PRINT TAB(20);"-mit vier Dezimalstellen ";TAB (6
0);"(4)"
270 PRINT TAB(20);"-im E-Format";TAB (60);"(5)"
280 PRINT TAB(20);"-so wie er eingegeben wird";TAB (
60);"(6)"
290 INPUT darstx
300 CLS
310 PRINT"Wie soll der y-Wert in der Tabelle darsges
tellt werden ?":PRINT:PRINT
```

```
320 PRINT TAB(20);"-mit einer Dezimalstelle   ";TAB
(60);"(1)"
330 PRINT TAB(20);"-mit zwei Dezimalstellen ";TAB (6
0);"(2)"
340 PRINT TAB(20);"-mit drei Dezimalstellen ";TAB (6
0);"(3)"
350 PRINT TAB(20);"-mit vier Dezimalstellen ";TAB (6
0);"(4)"
360 PRINT TAB(20);"-im E-Format";TAB (60);"(5)"
370 PRINT TAB(20);"-so wie er berechnet wird";TAB (6
0);"(6)"
380 INPUT darsty
390 CLS
400 PRINT#5,"Beenden der Eingabe durch Eingabe von
n ."
410 FOR k=1 TO 3
420 ORIGIN k*216-55,375 : DRAWR -130,0
430 ORIGIN k*216-120,400 : DRAWR 0,-32
440 ORIGIN k*216-121,400 : DRAWR 0,-32
450 ORIGIN k*216-119,400 : DRAWR 0,-32
460 LOCATE #k, 8,1 : PRINT#k,"X"
470 LOCATE #k, 16,1 : PRINT#k,"Y"
480 PRINT#k
490 FOR n=3 TO 22
500 INPUT#4,"X-Wert ";x$
510 CLS #4
520 IF x$="n" THEN GOTO 730
530 x=VAL(x$)
540 GOSUB 70 : REM Funktionsgleichung
550 ON darstx GOSUB 610,620,630,640,650,660 : REM Au
sdruck des x-Wertes
```

```
560 ON darsty GOSUB 670,680,690,700,710,720 : REM Au
sdruck des y-Wertes
570 DRAWR 0,-16 : DRAWR -1,0 : DRAWR 0,16 : DRAWR -1
,0 : DRAWR 0,-16 : DRAWR 2,0
580 NEXT n
590 NEXT k
600 GOTO 600
610 PRINT#k,USING "########.#";x; :RETURN
620 PRINT#k,USING "#######.##";x; :RETURN
630 PRINT#k,USING "######.###";x; :RETURN
640 PRINT#k,USING "#####.####";x; :RETURN
650 PRINT#k,USING "#.#####^^^^";x; :RETURN
660 PRINT#k,x;:RETURN
670 PRINT#k,TAB(14);ROUND (y,1) : RETURN
680 PRINT#k,TAB(14);ROUND (y,2) : RETURN
690 PRINT#k,TAB(14);ROUND (y,3) : RETURN
700 PRINT#k,TAB(14);ROUND (y,4) : RETURN
710 PRINT#k,TAB(14);USING"#.#####^^^^";y : RETURN

720 PRINT#k,TAB(14);y : RETURN
730 CLS#4
740 CLS#5
750 GOTO 750
760 REM Teilprogramm Einzelausgabe
770 CLS
780 WINDOW#1,1,80,10,15
790 PRINT#5,"Beenden der Eingabe durch Eingabe von n
."
800 INPUT#4,"X-Wert : ";x$
810 IF x$ = "n" THEN 870
820 x=VAL(x$)
```

```
830 GOSUB 70 : REM Funktionsgleichgung
840 PRINT#1,"X : ";x; TAB(20);"Y : ";y
850 CLS#4
860 GOTO 800
870 END
```

1.3 Berechnung von Funktionswerten nach dem Horner-Schema

Grundlagen

Will man den Funktionswert von ganzen rationalen Funktionen bestimmen, ist es oft vorteilhaft, ein Schema anzuwenden, das nach dem englischen Mathematiker William George HORNER (1786 bis 1837) benannt worden ist. Das Horner-Schema hat seine überragende Bedeutung allerdings verloren, seit diese Aufgabe mit Hilfe der Computer stark vereinfacht worden ist; muß man jedoch «zu Fuß» rechnen, ist es oft unmöglich, mit einigermaßen vertretbaren Rechenaufwand zu einem Ergebnis zu kommen, wenn man das Schema nicht anwendet. Da die Polynomdivision außerdem durch das Horner-Schema erleichtert wird (siehe Abschnitt 1.4), wird es an dieser Stelle besprochen.

Bei der Anwendung geht man folgendermaßen vor: Die Koeffizienten der ganzen rationalen Funktion

$$y = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0 \qquad (1.3.1)$$

werden nebeneinander geschrieben, wobei fehlende Glieder durch 0 zu berücksichtigen sind. In einer zweiten Zeile steht an erster Stelle eine 0. Die weiteren Spalten sowie die Glieder einer dritten Zeile werden wie folgt berechnet:

- ☐ die in der gleichen Spalte stehenden Zahlen der ersten und zweiten Zeile werden addiert und die Summe als c_n, c_{n-1}, ..., c_1, c_0 in Zeile drei eingetragen.
- ☐ diese Summe wird jeweils mit a, also mit dem Wert, für den der Funktionswert berechnet werden soll, multipliziert; das Ergebnis schreibt man um eine Zeile versetzt in die zweite Zeile.

Diese beiden Regeln werden abwechselnd angewendet, bis folgendes Schema entstanden ist:

$$\begin{array}{llllll} a_n & a_{n-1} & a_{n-2} & \dots & a_1 & a_0 \\ 0 & a \cdot c_n & a \cdot c_{n-1} & \dots & a \cdot c_2 & a \cdot c_1 \\ \hline c_n & c_{n-1} & c_{n-2} & \dots & c_1 & c_0 \end{array} \qquad (1.3.2)$$

Nach den oben genannten Regeln ist $c_n = a_n$; $c_{n-1} = a_{n-1} + a \cdot c_n$ usw. Der Wert c_0 schließlich stellt den gesuchten Funktionswert an der Stelle a dar.

Beispiel

Der Funktionswert der Funktion

$$y = x^5 - 2x^4 + 4x^2 + x - 5 \qquad (1.3.3)$$

soll an der Stelle $x = 2$ bestimmt werden. Als Horner-Schema ergibt sich:

$$\begin{array}{rrrrrr} 1 & -2 & 0 & 4 & 1 & -5 \\ 0 & 2 & 0 & 0 & 8 & 18 \\ \hline 1 & 0 & 0 & 4 & 9 & 13 \end{array}$$

Der Funktionswert an der Stelle $x = 2$ ist also $y = 13$.

Literaturhinweise

G: [5], S. 318 bis 320; G: [17], S. 43 bis 46; G: [24], S. 57 f.; G,W: [22], S. 210 f.; G,W: [21], S. 85 bis 92.

Programmbeschreibung

	Das Programm ist relativ kurz; es belegt nur 0,7 K im Speicher des CPC 464. In
Zeile 20	wird der Grad des Polynoms eingegeben (entspricht dem Wert n). Danach erfragt das Programm in den
Zeilen 40 bis 120	im Dialog die einzelnen Faktoren a_n bis a_0, wobei man fehlende Glieder durch Eingabe von 0 berücksichtigen muß. In

Zeile 130 wird schließlich der x-Wert eingegeben, für den die Berechnung durchgeführt werden soll (entspricht a). Das Horner-Schema läuft in den

Zeilen 140

bis 180 ab.

Zeile 190 gibt den Funktionswert aus, und die

Zeilen 200

bis 230 fragen ab, ob eine weitere Berechnung durchgeführt werden soll.

```
10 REM Programm Horner
20 CLS : INPUT"Grad des Polynoms : ";grad
30 DIM xges(grad+1,3)
40 PRINT:PRINT:PRINT"Geben Sie die Polynomfaktoren e
in ."
50 FOR k=grad TO 0 STEP -1
60 LOCATE 1,8 : PRINT"a";k;" : ";:INPUT xges(k,1)
70 LOCATE 6,8 : PRINT"                    "
80 NEXT k
90 PRINT:INPUT "Eingabe i.o. ( /n) ";a$
100 IF a$="" THEN 130
110 IF a$="n" THEN 40
120 GOTO 90
130 CLS : INPUT "x-Wert fuer die Funktionsberechnung
 : ";x
140 xges(grad,2)=0
150 FOR k=grad TO 0 STEP -1
160 xges(k,3)= xges(k,1) + xges(k,2) : IF k=0 THEN 1
80
170 xges(k-1,2)=xges(k,3)*x
180 NEXT k
190 CLS : PRINT"X = ";x; TAB(20);"Y = ";xges(0,3)
```

```
200 PRINT:PRINT:INPUT"Noch eine Berechnung ( /n) ";a
$
210 IF a$="" THEN 130
220 IF a$="n" THEN 240
230 GOTO 200
240 END
```

1.4 Polynomdivision

Jede ganze rationale Funktion $f(x)$ läßt sich als Produkt von sogenannten Linearfaktoren $(x - a_i)$ schreiben, wobei a_i jeweils eine Konstante darstellt. Ist a_i eine Nullstelle der Ausgangsfunktion, so entsteht nach dem Abspalten des Linearfaktors $(x - a_i)$, d.h. nach Division von $f(x)$ durch $(x - a_i)$, eine neue ganze rationale Funktion $f_a(x)$, deren Grad um 1 niedriger ist als der von $f(x)$. Spaltet man von $f_a(x)$ erneut eine Nullstelle ab, so ist der Grad der neuen Funktion wieder um 1 niedriger usw. Dieses kann man so lange wiederholen, bis die ursprüngliche Funktion ganz als Produkt von Linearfaktoren geschrieben wird. So hat z.B. die Funktion

$$f(x) = x^3 - 2x^2 - 15\,x \tag{1.4.1}$$

drei Nullstellen; sie liegen bei

$$x_{01} = 5; \quad x_{02} = 0; \quad x_{03} = -3$$

Somit kann man $f(x)$ auch wie folgt schreiben:

$$f(x) = (x - 5) * (x - 0) * (x + 3)$$

Durch Ausmultiplizieren kann man aus dieser Form wieder Gleichung (1.4.1) herstellen.

Die praktische Nutzanwendung der besprochenen Regel liegt z.B. in der Bestimmung von Nullstellen. Sind eine oder mehrere Nullstellen bekannt, so erhält man nach Abspalten der entsprechenden Linearfaktoren eine Funktion niedrigeren Grades, von der die restlichen Nullstellen leichter zu bestimmen sind. Sind z.B. von einer Funktion 4. Grades 2 Nullstellen bekannt, so erhält man eine Funktion 2. Grades, auf die man die bekannte Lösungsformel (siehe auch Abschnitt 1.7) anwenden kann.

Das Abspalten von Linearfaktoren kann man sich mit Hilfe des in Abschnitt 1.3 beschriebenen Horner-Schemas stark erleichtern. Man setzt lediglich als Wert a die bekannte Nullstelle ein und erhält die Koeffizienten der neuen Funktion direkt in der dritten Zeile (siehe hierzu auch [24], S. 58 f).

Hat die Ausgangsfunktion den Grad n, so ist der Faktor in der ersten Spalte der dritten Zeile der Koeffizient von x^{n-1}, der in der zweiten Spalte der Koeffizient von x^{n-2} usw. In der letzten Spalte muß 0 stehen, da ansonsten a keine Nullstelle darstellt.

Bezugnehmend auf das als Gleichung (1.3.2) aufgestellte allgemeine Schema erhält man für die Funktion

$$y = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0 \qquad (1.4.2)$$

demnach nach Abspalten des Linearfaktors $(x - a)$ die neue Funktion $f_a(x)$:

$$\begin{aligned} y &= f_a(x) * (x - a) \\ &= (c_n x^{n-1} + c_{n-1} x^{n-2} + \ldots + c_2 x + c_1) * (x - a), \end{aligned}$$

wenn a eine Nullstelle der Funktion darstellt.

Beispiel

Ausgangsfunktion: $f(x) = x^3 - 2x^2 - 15x$

Nullstelle: $x_{01} = 5$

Horner-Schema:

1	−2	−15	0
0	5	15	0
1	3	0	0

Also ergibt sich nach Abspalten des Linearfaktors $(x - 5)$ die neue Funktionsgleichung

$$f_a(x) = x^2 + 3x$$

Somit kann man die Ausgangsfunktion auch wie folgt schreiben:

$$f(x) = (x^2 + 3x) * (x - 5)$$

Die Gleichung $f_a(x)$ kann aber leicht gelöst werden; man erhält die Nullstellen

$$x_{01} = 0$$
$$x_{02} = -3$$

Literaturhinweise

G: [5], S. 318 bis 320; G: [17], S. 43 bis 46; G: [24], S. 57 f.; G,W: [22], S. 210 f.; G,W: [21], S. 85 bis 92

Programmbeschreibung

	Die
Zeilen 10 bis 190	entsprechen den ersten Zeilen des in Abschnitt 1.3 beschriebenen Programms Horner. Lediglich die nachgeschaltete Ausgabe ist anders. In den
Zeilen 200 bis 410	erfolgt die Ausgabe der Funktion und des Ergebnispolynoms mit dem eventuellem Rest (wenn hier ein von 0 verschiedener Wert auftaucht, war *a* keine Nullstelle). Über die
Zeilen 420 bis 440	kann vom Ergebnispolynom ein weiterer Linearfaktor abgespalten werden. In diesem Fall wird in den
Zeilen 480 bis 530	das Ergebnispolynom in den Speicherplatz für das Ausgangspolynom umgespeichert. Schließlich kann über die
Zeilen 450 bis 470	noch eine neue Rechnung durchgeführt werden.

Programmlänge: etwa 1,9 K.

```
10 REM Programm poldiv
20 REM Dieses Programm spaltet von einem Polynom unt
er Anwendung des Horner-Schemas einen Linearfaktor a
b und berechnet das Restpoly
nom. Dies kann beliebig oft wiederholt werden.
30 CLS : INPUT"Grad des Polynoms : ";grad
40 DIM xges(grad+1,3)
50 PRINT:PRINT:PRINT"Geben Sie die Polynomfaktoren e
in ."
60 FOR k=grad TO 0 STEP -1
70 LOCATE 1,8 : PRINT"a";k;" : ";:INPUT xges(k,1)
80 LOCATE 6,8 : PRINT"                  "
90 NEXT k
100 PRINT:INPUT "Eingabe i.o. ( /n) ";a$
110 IF a$="" THEN 140
120 IF a$="n" THEN 50
130 GOTO 100
140 CLS : INPUT "Nullstelle : ";x
150 xges(grad,2)=0
160 FOR k=grad TO 0 STEP -1
170 xges(k,3)= xges(k,1) + xges(k,2) : IF k=0 THEN 1
90
180 xges(k-1,2)=xges(k,3)*x
190 NEXT k
200 REM Ausgabe von Ursprungspolynom und Ergebnispol
ynom
210 CLS : PRINT"Ausgangspolynom : "
220 FOR i=grad TO 0 STEP -1
230 IF xges(i,1) = 0 THEN 300
240 IF xges(i,1) > 0 THEN PRINT"  + ";
250 PRINT xges(i,1);
```

```
260 IF i=0 THEN 300
270 IF i=1 THEN 290
280 PRINT"x^";i;: GOTO 300
290 PRINT"x";
300 NEXT i
310 PRINT:PRINT:PRINT"Berechnetes Polynom (Nullstell
e = ";x;")"
320 FOR i = grad TO 1 STEP -1
330 IF xges(i,3) = 0 THEN 400
340 IF xges (i,3) > 1 THEN PRINT" + ";
350 PRINT xges(i,3);
360 IF i-1=0 THEN 400
370 IF i-1=1 THEN 390
380 PRINT"x^";i-1;" ";: GOTO 400
390 PRINT"x ";
400 NEXT i
410 PRINT"        Rest : ";xges(0,3);"/( x -";x;")"

420 PRINT:PRINT:PRINT:PRINT"Soll das berechnete Poly
nom noch durch eine andere Nullstelle dividiert werd
en  ( /n) ?":PRINT:PRINT"Der
 Rest kann dabei nicht beruecksichtigt werden.":INPU
T a$
430 IF a$="" THEN 480
440 IF a$="n" THEN 450 ELSE 420
450 CLS : PRINT"Soll eine weitere Rechnung mit einem
 anderen Polynom durchgefuehrt werden ( /n)":INPUT a
$
460 IF a$="" THEN RUN 10
470 IF a$="n" THEN 540 ELSE 450
480 REM Umspeicherung des Polynoms
```

```
490 xges(grad,1) = 0
500 FOR i=grad-1 TO 0 STEP -1
510 xges(i,1) = xges(i+1,3)
520 NEXT i
530 GOTO 140
540 END
```

1.5 Berechnung der Nullstellen einer Funktion mit Hilfe der Regula Falsi

Grundlagen

Nullstellenbestimmungen nach der Regula Falsi gehören zu den Näherungsverfahren, die dann angewendet werden, wenn eine Gleichung nicht oder nur mit erheblichem Aufwand geschlossen lösbar ist. Wie der Name sagt, erhält man mit einem solchen Verfahren nur einen Näherungswert für die tatsächliche Nullstelle. Normalerweise wird iterativ vorgegangen, so daß man durch Erhöhung der Rechenschrittzahl sehr hohe Genauigkeiten erreichen kann. Bei Verwendung eines Computers, bei dem ja die Anzahl der Rechenschritte praktisch keine Rolle spielt, erhält man oft den genauen Wert der Nullstelle.

Bei Anwendung der Regula Falsi müssen zwei Näherungswerte x_1 und x_2 bekannt sein; für x_1 muß der Funktionswert $f(x_1)$ kleiner 0 sein, für x_2 muß $f(x_2)$ größer 0 sein. Bei einer steigenden Kurve liegt x_1 also links von der Nullstelle und x_2 rechts; bei einer fallenden Kurve ist es umgekehrt.

Die Werte von x_1 und x_2 kann man z.B. aus einer Wertetabelle (s. Abschn. 1.2) oder aus der Zeichnung des Graphen der Funktion (s. Abschn. 1.1) entnehmen. Wenn die beiden Näherungswerte x_1 und x_2 bzw. die dazugehörigen Funktionspunkte $P_1(x_1, y_1)$ und $P_2(x_2, y_2)$ bekannt sind, werden P_1 und P_2 durch eine Gerade verbunden. Diese Gerade stellt eine Sehne der Funktion dar; daher wird die Regula Falsi auch oft als Sehnen-Verfahren bezeichnet.

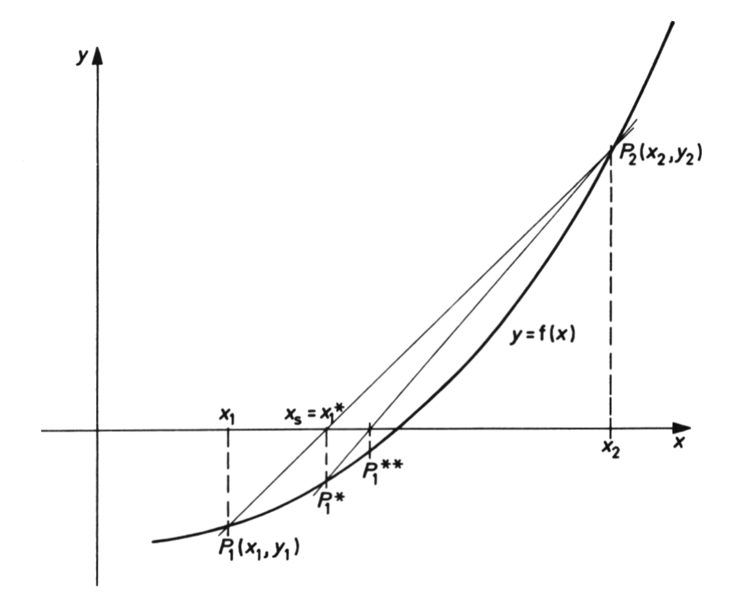


Bild 1.1 Konvergierende Regula Falsi

Die Gleichung der Sehne lautet:

$$\frac{y - y_1}{x - x_1} = \frac{y_2 - y_1}{x_2 - x_1} \tag{1.5.1}$$

x = beliebiger x-Wert
y = dazugehöriger y-Wert

Gleichung (1.5.1) sagt nichts anderes aus, als daß die Steigung der Sehne konstant ist.

Wenn $x = x_S$ ist (siehe Bild 1.1), gilt für die Sehne: $y = 0$. Somit erhält man für den Schnittpunkt der Sehne mit der x-Achse aus Gleichung (1.5.1):

$$x_S = x_1 - y_1 \frac{x_2 - x_1}{y_2 - y_1} \tag{1.5.2}$$

Wenn man jetzt $x_S = x_1^*$ setzt, mit diesem Wert aus der zu untersuchenden Funktionsgleichung den dazugehörigen Funktionswert $f(x_1^*)$ berechnet, diesen als y_1 und x_1^* als x_1 in Gleichung (1.5.2) einsetzt und somit eine neues x_S berechnet, stellt man fest, daß x_S näher an die Nullstelle heranwandert. Daher wiederholt man das Verfahren so lange, bis die erforderliche Genauigkeit erreicht ist.

Wenn man P_1 und P_2 umgekehrt wählt, daß also $f(x_1)$ größer 0 und $f(x_2)$ kleiner 0 ist, konvergiert das Verfahren nicht, wie in Bild 1.2 dargestellt ist. Die erste Sehne ist die gleiche wie in Bild 1.1; setzt man dann jedoch $s_S = x_1$ und verfährt wie oben angegeben, liegt der Schnittpunkt der neuen Sehne mit der x-Achse rechts von der Nullstelle; man ist also «übers Ziel hinausgeschossen». Wählt man P_1 und P_2 jedoch wie ursprünglich gesagt, kann man das Verfahren auf jede beliebige Funktion anwenden, egal, ob rechts- oder linksgekrümmt, ob steigend oder fallend.

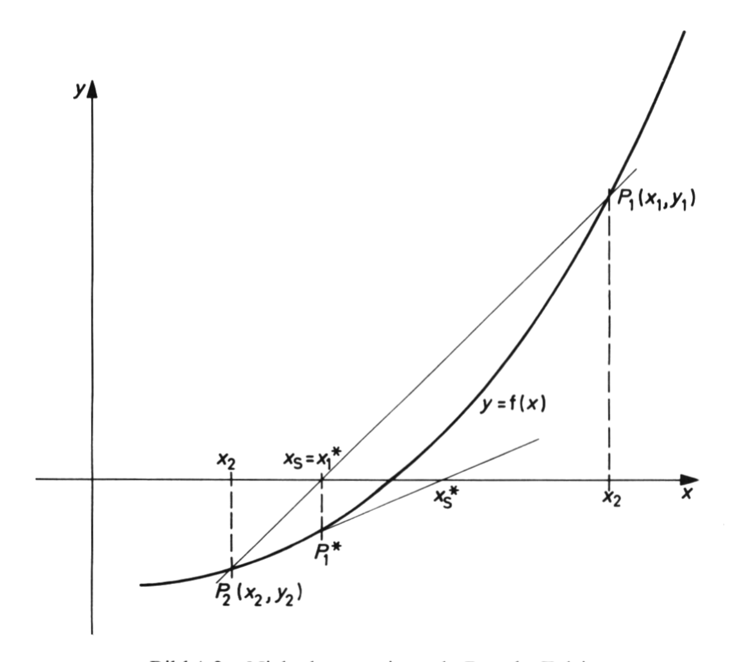


Bild 1.2 Nicht konvergierende Regula Falsi

Beispiel

Für die Funktion

$$y = x^2 - 9$$

soll die Nullstelle gesucht werden, die zwischen 2,5 und 3,5 liegt, und zwar mit einer maximalen Abweichung von 0,1 bezogen auf den y-Wert.

$$x = 2{,}5 \qquad f(x) = -2{,}75$$
$$x = 3{,}5 \qquad f(x) = 3{,}25$$

Daraus folgt die Festlegung:

$$x_1 = 2{,}5 \qquad y_1 = -2{,}75$$
$$x_2 = 3{,}5 \qquad y_2 = 3{,}25$$

Aus Gleichung (1.5.2) erhält man für den Schnittpunkt der Sehne mit der x-Achse

$$x_S = 2{,}5 + 2{,}75 \frac{3,\ 5 - 2{,}5}{3{,}25 + 2{,}75}$$
$$= 2{,}958$$
$$= x_1^*$$

Hierfür erhält man den Funktionswert

$$f(x_1^*) = 2{,}958^2 - 9$$
$$= -0{,}25$$

Erneut eingesetzt in Gleichung (1.5.2) erhält man

$$x_S = 2{,}958 + 0{,}25 \frac{3{,}5 - 2{,}958}{3{,}25 + 0{,}25}$$
$$= 2.997$$

Der Funktionswert hierfür ist $-0{,}018$, so daß die geforderte Genauigkeit bereits in zwei Schritten erreicht ist.

Literaturhinweise

G: [4], S. 314; G,W: [5], S. 684 bis 688.

Programmbeschreibung

Auch das Programm *Regula Falsi* ist sehr kurz; Es hat eine Länge von 0,9 K. In

Zeile 50 ist die zu untersuchende Funktionsgleichung abgelegt. In den

Zeilen 70 bis 110 erfolgt die Eingabe der Schätzwerte sowie der Genauigkeitsgrenze. Die Größe n in

Zeile 120 stellt einen Zähler dar, mit der die Anzahl der Durchläufe festgestellt wird, die der Rechner benötigt, um die geforderte Genauigkeit zu erreichen. In den

Zeilen 130 und 140 werden für beide Schätzwerte die entsprechenden Funktionswerte berechnet; die Regula-Falsi-Formel läuft in

Zeile 150 ab. In

Zeile 160 erfolgt die Prüfung, ob die gezielte Genauigkeit ausreichend ist; falls nicht, wird eine neue Rechnung durchgeführt, nachdem der Zähler n um 1 erhöht wurde. Reicht die Genauigkeit dagegen, wird die Nullstelle mit der Anzahl der benötigten Durchläufe sowie dem Funktionswert ausgegeben. In den

Zeilen 200 und 210 wird schließlich abgefragt, ob noch eine weitere Nullstelle bestimmt werden soll.

```
10 REM  Programm Regula Falsi
20 MODE 2
30 GOTO 70
40 REM Zeile 50 = Funktionsgleichung
50 y = x*x - 9
60 RETURN
70 CLS
80 INPUT "Schaetzwert 1 fuer x0  (f(x0) < 0 !!!!) ";
x1
90 PRINT:INPUT "Schaetzwert 2 fuer x0  (f(x0) > 0 !!
!!) ";x2
100 PRINT:PRINT:PRINT"Mit welcher Genauigkeit soll d
ie Nullstelle Berechnet werden ? "
110 PRINT:INPUT "Geben Sie die maximale Abweichung a
n ";grenze
120 n=1 : REM Durchlaufzaehler
130 x=x1 : GOSUB 50 : y1=y
140 x=x2 : GOSUB 50 : y2=y
150 xs = x1 - y1*((x2-x1)/(y2-y1))
160 x=xs : GOSUB 50 : IF ABS(y) < ABS(grenze) THEN 1
90
170 x1=xs : y1=y
180 n=n+1 : GOTO 150
190 CLS: PRINT"x = ";xs;TAB(20);"y = ";y
200 PRINT:PRINT"Zur Berechnung wurden ";n;"Durchlaeu
fe benoetigt"
210 PRINT:PRINT"Noch eine Nullstellenberechnung derg
leichen Funktion ( /n) ";:INPUT a$
220 IF a$ = "" THEN 70
230 IF a$ = "n" THEN END
240 GOTO 210
```

1.6 Berechnung der Nullstellen einer beliebigen Funktion

In diesem Abschnitt wird ein Programm beschrieben, das eine ganz typische Rechneranwendung darstellt, da eine Vielzahl von Rechnungen ausgeführt werden, die von Hand mit vertretbarem Zeitaufwand gar nicht durchgeführt werden könnten.

Das Programm berechnet die Nullstelle einer gegebenen Funktion, indem die Kurve einfach mit einer sehr kleinen Schrittweite abgetastet wird. Wechselt das Vorzeichen des jeweiligen Funktionswertes, wird die Schrittweite halbiert und die Kurve in umgekehrter Richtung abgetastet.

Dieses Verfahren wird so lange angewendet, bis die erforderliche Genauigkeit erreicht ist. Von der gefundenen Nullstelle geht es dann mit der ursprünglichen Schrittweite weiter.

Gegenüber dem in Abschnitt 1.5 beschriebenen Regula-Falsi-Verfahren hat diese Methode vor allen Dingen den Vorteil, daß keine Schätzwerte für die Nullstelle benötigt werden.

	Die Funktionsgleichung ist in
Zeile 70	abgelegt. Die
Zeilen 90	
bis 140	dienen der Eingabe der Intervallgrenzen, der Schrittweite und der erforderlichen Genauigkeit.
	Das Abtasten der Funktion erfolgt in den
Zeilen 150	
bis 230	wobei in
Zeile 210	der beschriebene Vorzeichenwechsel stattfindet. In
Zeile 240	erfolgt die Ausgabe der gefundenen Nullstelle, und über die
Zeilen 250	
bis 260	wird die Suche fortgesetzt, nachdem eine Nullstelle gefunden worden ist.
	Länge des Programms: etwa 1 K.

```
10 REM Programm NullRechner
20 REM Die Nullstelle wird berechnet, indem die Kurv
e mit vorgegebener Schritt- weite abgetastet wird, b
ei Vorzeichenwechsel die Sch
```

rittweite halbiert wird etc.

```
30 ON ERROR GOTO 40
40 RESUME NEXT
50 GOTO 90
60 REM Zeile 60 = Funktionsgleichung
70 y=-2*x*x*x*x*x + 10*x*x*x - 8*x
80 RETURN
90 CLS : PRINT"Geben Sie den Startwert der Nullstell
ensuche ein."
100 INPUT"Es wird in positiver Richtung gesucht!";xs
110 PRINT : PRINT"Bis zu welchem x-Wert soll die Nul
lstellensuche fortgesetzt werden ";:INPUT xe
120 PRINT:INPUT"Mit welcher Schrittweite soll vorgeg
angen werden ";sw0
130 PRINT:PRINT"Mit welcher Genauigkeit soll die Nul
lstelle berechnet werden ?"
140 INPUT"Geben Sie die maximale Abweichung ein. ";g
renze
150 x=xs : GOSUB 70 : y1=y
160 k=2 : sw = sw0
170  x=x+sw
180 IF x>= xe THEN 270
190 GOSUB 70
200 y2=y : IF ABS(y) < ABS(grenze) THEN 240
210 IF SGN(y1)=SGN(y2) THEN 220 ELSE 230
220 y1=y2 : GOTO 170
230 sw=-sw/k : y1=y2 : GOTO 170
240 PRINT:PRINT"x0 = ";x,"y = ";y
250 x=x+sw0 : GOSUB 70
260 sw = sw0 : y1=y : GOTO 170
270 END
```

1.7 Lösen quadratischer Gleichungen

Grundlagen

Die allgemeine Form einer quadratischen Gleichung lautet

$$a_2 x^2 + a_1 x + a_0 = 0 \tag{1.7.1}$$

Mittels Division durch a_2 führt man diese Gleichung in die sogenannte Normalform über:

$$x^2 + \frac{a_1}{a_2} x + \frac{a_0}{a_2} = 0 \tag{1.7.2}$$

Durch Termersetzung erhält man

$$x^2 + px + q = 0 \qquad \text{bzw.}$$

$$x^2 + px = -q \tag{1.7.3}$$

Zu dieser Gleichung addiert man auf beiden Seiten die quadratische Ergänzung $(p/2)^2$.

$$x^2 + px + \left(\frac{p}{2}\right)^2 = -q + \left(\frac{p}{2}\right)^2$$

Somit kann man aber nach dem binomischen Lehrsatz die linke Seite folgendermaßen umformen:

$$\left(x + \frac{p}{2}\right)^2 = \left(\frac{p}{2}\right)^2 - q$$

Jetzt wird auf beiden Seiten die Wurzel gezogen.

$$\sqrt{\left(x + \frac{p}{2}\right)^2} = \sqrt{\left(\frac{p}{2}\right)^2 - q}$$

$$\pm \left(x + \frac{p}{2}\right) = \sqrt{\left(\frac{p}{2}\right)^2 - q}$$

Hieraus erhält man die beiden bekannten Lösungsformeln:

$$x_1 = -\frac{p}{2} + \sqrt{\left(\frac{p}{2}\right)^2 - q} \tag{1.7.4}$$

$$x_2 = -\frac{p}{2} - \sqrt{\left(\frac{p}{2}\right)^2 - q} \qquad (1.7.5)$$

Zur Lösung einer quadratischen Gleichung stellt man also zunächst die Normalform her und setzt dann die Faktoren p und q in Gleichung (1.7.4) und (1.7.5) ein.

Beispiel

$$4x^2 + 16x - 48 = 0$$

Normalform: $x^2 + 4x - 12 = 0$

Hieraus folgen direkt die Lösungen:

$$x_1 = -2 + \sqrt{2^2 + 12} = 2$$

$$x_2 = -2 - \sqrt{2^2 + 12} = -6$$

Literaturhinweise

G: [15], S. 145 bis 162; G: [5], S. 251 ff.; G: [6], S. 69 bis 72

Allgemeine Literatur zu Kapitel 1

G: [15], S. 107 ff.; G: [5], S. 224 bis 225; G: [12], S. 176 ff.; G,W: [22], S. 179 ff.; G,W: [6], S. 83 ff.; G,W: [24], S. 27 ff.

Programmbeschreibung

Zeile 20
bis 110 Eingabe der Faktoren a_2, a_1 und a_0

Zeilen 120
und 130 Termersetzung

Zeile 140 Für den Fall, daß der Wert unter der Wurzel kleiner 0 ist, erfolgt keine Berechnung, sondern der Sprung nach
Zeile 250

Zeile 150
und 160 Berechnung der Lösungen

Zeile 170 Ausgabe

Zeile 180
bis 200 Abfrage, ob eine weitere Rechnung durchgeführt werden soll.
Programmlänge: etwa 0,8 K.

```
10 REM Programm Quadratgleich
20 CLS : PRINT"Geben Sie die Faktoren der quadratisc
hen Gleichung ein ."
50 PRINT:PRINT"y = a2*x^2 + a1*x + a0"
60 LOCATE 1,6 : INPUT "a2 : ";a2
70 INPUT "a1 : ";a1
80 INPUT"a0 : ";a0
90 PRINT:INPUT "Eingaben i.o. ( /n) ";a$
100 IF a$ = "" THEN 120
110 IF a$ = "n" THEN 10 ELSE 90
120 p=a1/a2
130 q=a0/a2
140 IF (p*p/4 - q) < 0 THEN 250
150 x01 = -p/2 + SQR(p*p/4 - q)
160 x02 = -p/2 - SQR(p*p/4 - q)
170 PRINT:PRINT:PRINT"x01 = ";x01 : PRINT "x02 = ";x
02
180 PRINT:PRINT"Noch eine Gleichung ( /n) ";:INPUT a
$
190 IF a$="" THEN 10
200 IF a$="n" THEN END ELSE 180
250 PRINT:PRINT"Die Gleichungen hat imaginaere Loesu
ngen. Loesung leider nicht moeglich.":GOTO 180
```

2
Differentialrechnung

2.1 Bestimmung der Ableitung von rationalen Funktionen

Grundlagen

Eine ganze rationale Funktion hat die Form

$$y = a_1 x^{n_1} + a_2 x^{n_2} + \ldots + a_m x^{n_m} \qquad (2.1.1)$$

Die 1. Ableitung dieser Funktion berechnet sich wie folgt (s. auch [6], S. 198 ff):

$$y' = a_1 n_1 x^{n_1 - 1} + a_2 n_2 x^{n_2 - 1} + \ldots + a_m n_m x^{n_m - 1} \qquad (2.1.2)$$

Die 2. Ableitung erhält man, indem man die gleiche Regel auf die Funktionsgleichung der 1. Ableitung anwendet. Der Grad von x verringert sich beim Bilden der Ableitung also um 1. Hieraus folgt, daß die Ableitung einer Geraden gleich einer Konstanten ist:

$$y = a_1 x^1$$

$$y' = a_1 \cdot 1 \cdot x^0 = a_1$$

Die Ableitung einer Konstanten ist dagegen gleich 0, da die Ableitung einer Funktion ja gleichzusetzen ist mit der Steigung der Funktion; eine Konstante ist aber eine Parallele zur x-Achse und hat somit keine Steigung. Das gleiche Ergebnis erhält man bei «sturer» Anwendung der Ableitungsregel:

$$y = a_0 = a_0 \cdot x^0$$

$$y' = 0 \cdot a_0 \cdot x^{-1} = 0$$

Auch auf gebrochene Exponenten von x läßt sich obige Regel anwenden:

$$y = a_1 \cdot x^{1/2} = a_1 \cdot \sqrt[2]{x}$$

$$y' = {}^1/_2 \cdot a_1 \cdot x^{-1/2} = \frac{a_1}{2 \cdot \sqrt{x}}$$

Beispiel

$$\begin{aligned} y &= 5x^4 - 2x^2 + 3 \\ y' &= 20x^3 - 4x \\ y'' &= 60x^2 - 4 \\ y''' &= 120x \end{aligned}$$

Programmbeschreibung

Die Funktion wird per Dialog in den

Zeilen 30
bis 140 eingeben. Auch bei Konstanten muß hierbei der Exponent von x eingegeben werden (er ist hier gleich 0).
Die Berechnung der Ableitungen erfolgt in den

Zeilen 180
bis 250 Die restlichen Zeilen dienen lediglich der Ausgabe. Die Funktion selbst wird in den

Zeilen 280
bis 370 ausgegeben.

Zeile 300 unterdrückt hierbei die Ausgabe des jeweiligen Gliedes, falls der Faktor vor x gleich 0 ist, da das Glied in diesem Fall nicht vorhanden ist.

Zeile 310 druckt ein + aus, wenn dies erforderlich ist. Ein – wird automatisch ausgegeben.

Zeile 320 gibt den Faktor aus.
Falls der Exponent von x gleich 0 ist, erfolgt mittels

Zeile 330 sofort die Ausgabe des nächsten Gliedes.

Zeile 350 druckt dagegen ein zusätzliches x aus, wenn der Exponent gleich 1 ist. Trifft beides nicht zu, wird über

Zeile 340 bzw.

Zeile 360	ein x und der dazugehörige Exponent ausgegeben. Der beschriebene Vorgang wird in den
Zeilen 380	
bis 670	für die 1. bis3. Ableitung wiederholt; und schließlich kann das Programm über die Abfrage in
Zeile 680	neu gestartet werden.
	Programmlänge: etwa 2,3 K.

```
10 REM Programm polableit
20 REM Mit diesem Programm koennen von ganzen ration
alen Funktionen die ersten drei Ableitungen berechne
t werden
30 MODE 2 : PRINT"Untersucht wird eine ganze rationa
le Funktion der Form"
40 PRINT:PRINT"y = a1*x^n1 + a2*x^n2 + ... + am*x^nm
"
50 PRINT:INPUT "Wie viele Glieder hat die Funktion "
;m
60 DIM ao(m),n(m)
70 PRINT:PRINT"Geben Sie die einzelnen Faktoren ein.
"
80 FOR i=1 TO m
90 LOCATE 1,10 : PRINT "a";i;" : ";:INPUT ;aO(i):PRI
NT TAB (30); "n";i;" : ";:INPUT nO(i)
100 LOCATE 1,10 : PRINT"
                       "
110 NEXT i
120 PRINT:INPUT "Eingaben alle i.o. ( /n) ";a$
130 IF a$="" THEN 150
140 IF a$="n" THEN RUN 10 ELSE 120
150 DIM a1(m),n1(m) : REM 1.Ableitung
```

```
160 DIM a2(m),n2(m) : REM 2.Ableitung
170 DIM a3(m),n3(m) : REM 3.Ableitung
180 FOR i=1 TO m
190 a1(i) = a0(i) * n0(i)
200 n1(i) = n0(i) - 1
210 a2(i) = a1(i) * n1(i)
220 n2(i) = n1(i) - 1
230 a3(i) = a2(i) * n2(i)
240 n3(i) = n2(i) - 1
250 NEXT i
260 REM konstanter Faktor wird automatisch beruecksi
chtigt
270 REM Ausgabe der Gleichungen mit den Unbekannten
280 CLS : PRINT"Gleichung "
290 FOR i=1 TO m
300 IF a0(i) = 0 THEN 370
310 IF a0(i) > 0 THEN PRINT" + ";
320 PRINT a0(i);
330 IF n0(i) = 0 THEN 370
340 IF n0(i) <> 1 THEN 360
350 PRINT"x ";:GOTO 370
360 PRINT "x^";n0(i);" ";
370 NEXT i
380 PRINT:PRINT:PRINT"1.Ableitung":
390 FOR i=1 TO m
400 IF a1(i) = 0 THEN 470
410 IF a1(i) > 0 THEN PRINT" + ";
420 PRINT a1(i);
430 IF n1(i) = 0 THEN 470
440 IF n1(i) <> 1 THEN 460
```

```
450 PRINT"x ";: GOTO 470
460 PRINT"x^";n1(i);" ";
470 NEXT i
480 PRINT:PRINT:PRINT"2.Ableitung"
490 FOR i=1 TO m
500 IF a2(i) = 0 THEN 570
510 IF a2(i) > 0 THEN PRINT" + ";
520 PRINT a2(i);
530 IF n2(i) = 0 THEN 570
540 IF n2(i) <> 1 THEN 560
550 PRINT"x ";:GOTO 570
560 PRINT"x^";n2(i);" ";
570 NEXT i
580 PRINT:PRINT:PRINT"3.Ableitung"
590 FOR i=1 TO m
600 IF a3(i) = 0 THEN 670
610 IF a3(i) > 0 THEN PRINT" + ";
620 PRINT a3(i);
630 IF n3(i) = 0 THEN 670
640 IF n3(i) <> 1 THEN 660
650 PRINT"x ";:GOTO 670
660 PRINT"x^";n3(i);" ";
670 NEXT i
680 PRINT:PRINT:PRINT:PRINT"Soll noch eine Gleichung
 untersucht werden ( /n) ";:INPUT a$
690 IF a$ = "" THEN RUN 10
700 IF a$ = "n" THEN 710 ELSE 680
710 END
```

2.2 Nullstellenbestimmung nach Newton

Grundlagen

Genau wie die in Abschnitt 1.5 behandelte Regula Falsi handelt es sich beim Newton-Verfahren (Bild 2.1) um ein Näherungsverfahren zur Nullstellenbestimmung. Es hat den Vorteil, daß nur ein Schätzwert benötigt wird, der sowohl rechts als auch links von der Nullstelle liegen kann. Allerdings muß die Ableitung der Funktion bekannt sein.

Der Funktionspunkt für den ersten Schätzwert x_1 ist $P_1(x_1, y_1)$. In diesem Punkt legt man eine Tangente an die Kurve und bestimmt den Schnittpunkt der Tangente mit der x-Achse.

Die Tangente im Punkt P_1 hat die gleiche Steigung m wie die Funktion selbst in P_1; diese ist aber identisch mit der 1. Ableitung der Funktion $f'(x_1)$ für den x-Wert x_1. Somit ist man in der Lage, die Gleichung der Tangente aufzustellen (vgl. Abschnitt 1.5):

$$\frac{y - y_1}{x - x_1} = m = f'(x_1) \tag{2.2.1}$$

$x =$ beliebiger x-Wert
$y =$ dazugehöriger y-Wert

Für den Schnittpunkt der Tangente mit der x-Achse $x = x_S$ gilt wiederum: $y = 0$.

Daraus folgt:

$$x_S = x_1 - \frac{y_1}{f'(x_1)} \tag{2.2.2}$$

Analog zu Abschnitt 1.5 wird hierfür der Funktionswert der zu untersuchenden Gleichung berechnet, die zu x_S gehörige Ableitung der Funktion bestimmt und damit dann über Gleichung (2.2.2) der nächste Wert für x_S berechnet. Dieses Verfahren wird so oft angewendet, bis die erforderliche Genauigkeit erreicht ist; die berechneten Werte sind immer wieder einzusetzen in die Formel

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} \tag{2.2.3}$$

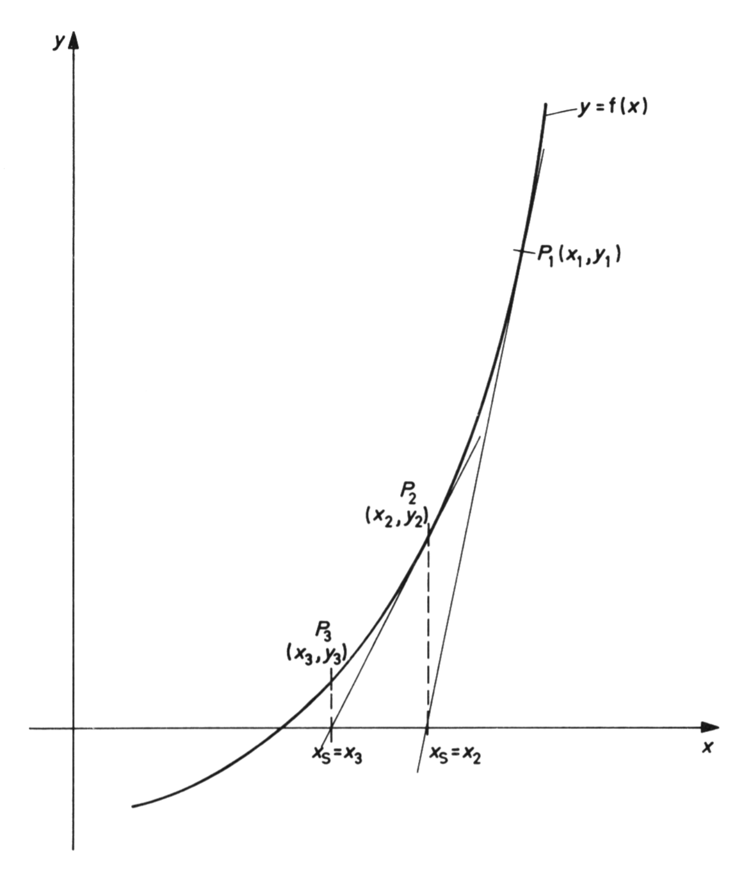


Bild 2.1 Newton-Verfahren

Beispiel

Von der Funktion

$$y = x^3 - 7x + 12$$

ist die Nullstelle in der Nähe von $x_1 = -3$ zu bestimmen, und zwar mit einer Genauigkeit von 0,001 bezogen auf den y-Wert.

$$f'(x) = 3x^2 - 7$$

$$f(x_1) = -3 + 7*3 + 12 = 6$$

$$f'(x_1) = 3*3^2 - 7 = 20$$

In Gleichung (2.2.3) eingesetzt, ergibt sich

$$x_2 = -3 - \frac{6}{20} = -3{,}3$$

Hierfür berechnet sich als Funktionswert $f(x_2) = -0{,}837$, so daß ein neuer Rechenschritt notwendig wird.

$$f'(x_2) = 25{,}67$$

$$x_3 = -3{,}3 - \frac{0{,}837}{25{,}67}$$

$$= -3{,}267$$

Für diesen Wert ist der Funktionswert $f(x_3) = -6{,}35*10^{-4}$ so daß die Genauigkeit ausreicht.

Literaturhinweise

G: [6], S. 196; G: [4], S. 312 bis 314; G, W: [21], S. 74 ff.

Programmbeschreibung

Zur Nullstellenbestimmung nach Newton müssen sowohl die Funktionsgleichung als auch die 1. Ableitung dem Programm bekannt sein; beide sind in

Zeile 60 bzw.

Zeile 90 abgelegt. Durch

Zeile 110
und 120 werden Schätzwert und Genauigkeitsanforderung dem Programm mitgeteilt; die eigentliche Rechnung wird in den
Zeilen 140
bis 200 durchgeführt.
Zeile 220 dient der Ausgabe, und über
Zeile 230
bis 250 kann eine weitere Nullstelle gesucht werden.

```
10 REM Programm Nullnewton
20 REM Programm zur Nullstellenbestimmung mit dem Ve
rfahren nach Newton
30 GOTO 110
40 REM In der Zeile 60 muss die Funktionsgleichung,
in der Zeile 90 die 1.Ableitung abgespeichert sein

50 REM Funktionsgleichung
60 y = x*x*x - 4*x*x - 11*x + 30
70 RETURN
80 REM 1.Ableitung
90 y = 3*x*x - 8*x - 11
100 RETURN
110 CLS : PRINT"Mit welcher Genauigkeit soll(en) die
 Nullstelle(n) berechnet werden ?           Geben Si
e die maximale Abweichung ei
n .";:INPUT abw
120 CLS : PRINT"Geben Sie einen Schaetzwert fuer die
 Nullstelle ein .";:INPUT x
130 REM so, jetzt geht's los (oder wieder los)
140 GOSUB 60 : REM Funktionswert fuer x
150 IF ABS(y) <= ABS(abw) THEN 210
160 yi = y
```

```
170 GOSUB 90 : REM 1. Ableitung fuer x
180 yi1 = y
190 x = x - yi/yi1 : REM neuer Naeherungswert
200 GOTO 130
210 REM wow, geschafft!
220 PRINT"x0 = ";x,"y = ";y
230 PRINT:PRINT"Noch eine Nullstelle bei derselben G
leichung ( /n) ";:INPUT a$
240 IF a$ = "" THEN 120
250 IF a$ = "n" THEN END ELSE 230
```

2.3 Extremwerte von Funktionen

Grundlagen

Die Extremwerte einer Funktion (Minimum bzw. Maximum) sind dadurch gekennzeichnet, daß die Funktion in diesem Punkt waagerecht verluft (Bild 2.2 und 2.3).

Legt man im Minimum bzw. Maximum eine Tangente an die Kurve, so ist diese eine Parallele zur x-Achse; die Steigung der Tangente ist gleich 0. Da Steigung von Tangente und Funktion per Definition aber identisch sind, gilt, daß auch die Steigung der Funktion gleich 0 ist.

Dies heißt aber mit anderen Worten, daß die 1. Ableitung in diesem Punkt gleich 0 ist (s. auch [6], S. 207 ff).

Um den Extremwert einer Funktion zu bestimmen, muß man also lediglich eine Nullstelle der 1. Ableitung suchen. Dies reicht jedoch nicht aus, um darüber zu entscheiden, ob ein Minimum oder ein Maximum der Funktion vorliegt. Um hierüber eine Aussage treffen zu können, muß man sich den Verlauf der 2. Ableitung vergegenwärtigen, d.h. den Verlauf der Steigung der 1. Ableitung (Bild 2.4).

Im Bild 2.4 sind mit steigendem x-Wert Tangenten an eine Kurve mit Maximum gelegt. Wie man sieht, nimmt die Steigung der Tangenten, also auch die der Funktion, mit steigendem x-Wert bei Annäherung an das Maximum ab; die Tangenten werden immer flacher, bis im Maximum eine waagerechte Tangente vorliegt. Mit anderen Worten heißt das, daß

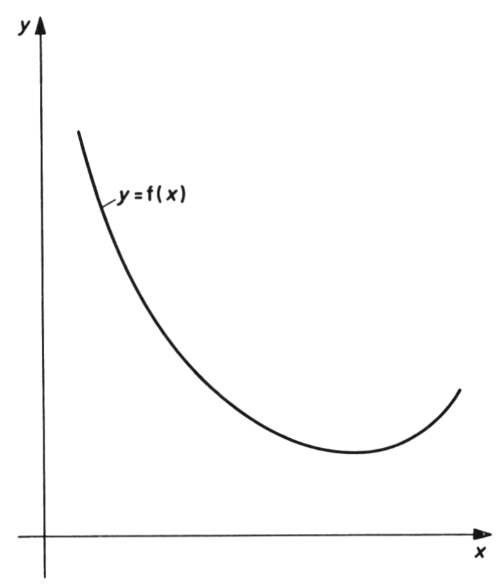


Bild 2.2 Funktion mit Minimum

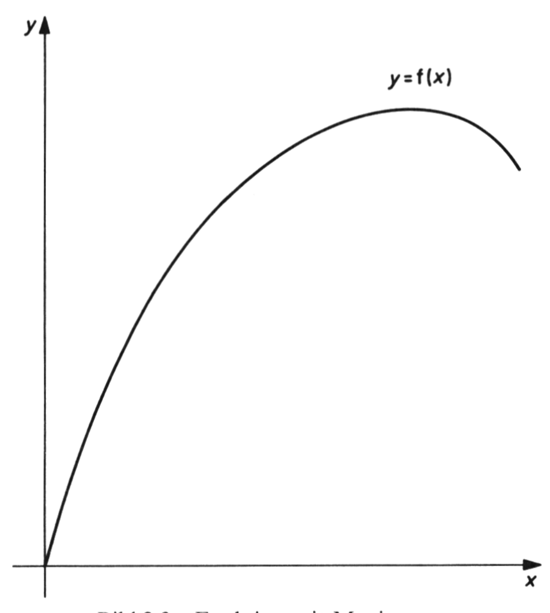


Bild 2.3 Funktion mit Maximum

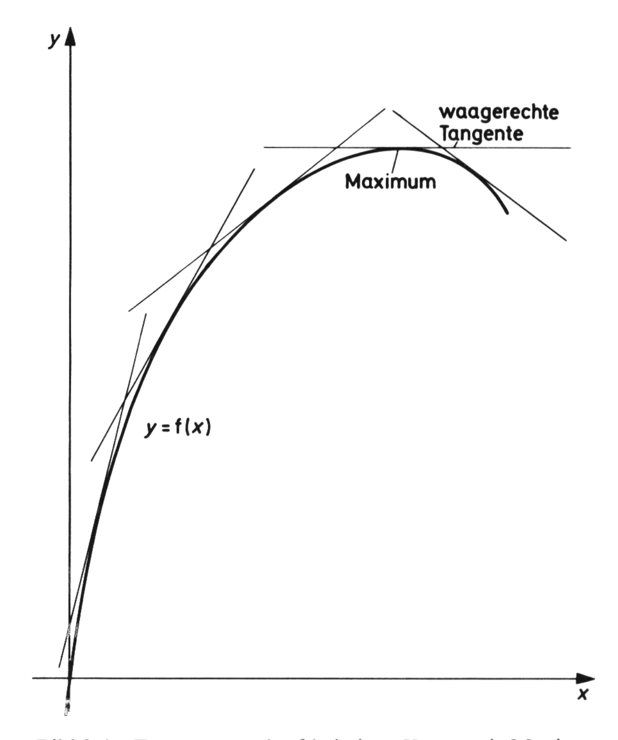


Bild 2.4 Tangentenverlauf bei einer Kurve mit Maximum

die 1. Ableitung der Funktion immer kleiner wird, bis sie im Maximum zu 0 wird, um dann in den negativen Bereich zu gehen (die Tangenten kippen zur anderen Seite). Wenn die 1. Ableitung jedoch fällt (kleiner wird), heißt das, daß ihre Steigung negativ ist oder daß die 2. Ableitung kleiner 0 ist.

Zusammenfassend kann man sagen, daß folgende Bedingungen erfüllt sein müssen, wenn ein Maximum vorliegt:

Maximum: 1. Ableitung $= 0$
2. Ableitung < 0

In gleicher Weise kann man die Bedingungen für ein Minimum beweisen:

Minimum: 1. Ableitung $= 0$
2. Ableitung > 0

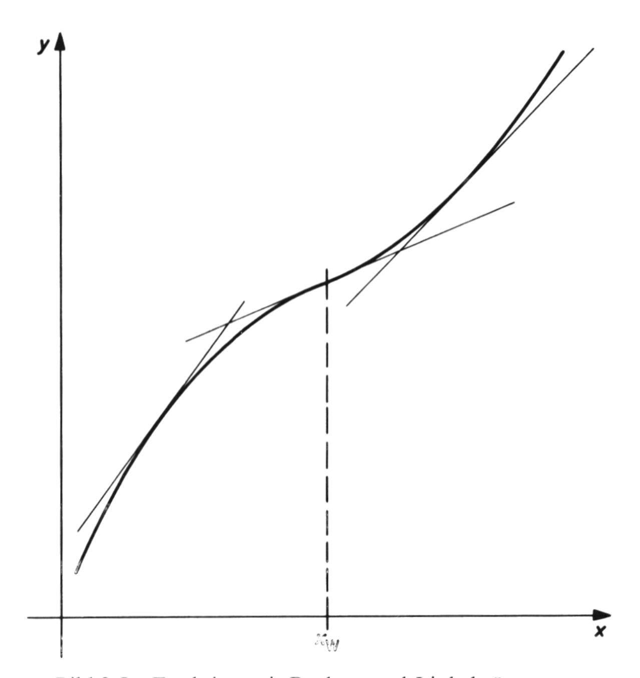


Bild 2.5 Funktion mit Rechts- und Linkskrümmung

Um den Krümmungsverlauf einer Kurve zu untersuchen, muß man ähnliche Betrachtungen anstellen:

Die Funktion in Bild 2.5 beschreibt zunächst eine Rechtskurve, die Steigung der Tangente bzw. Kurve oder der Wert der 1. Ableitung nimmt ab. Von x_w an aufwärts jedoch beschreibt die Funktion eine Linkskurve, d.h., ihr Drehsinn ist dann nach links gerichtet. In x_w geht die 1. Ableitung also von einer fallenden in eine steigende Bewegung über; die Steigung der 1. Ableitung (gleichbedeutend mit der 2. Ableitung) geht demnach aus dem negativen Bereich (= fallende Bewegung) in den positiven Bereich (= steigende Bewegung) über, in x_w geht sie durch 0. Für den Wechsel von einer Rechtskrümmung in eine Linkskrümmung (= Wendestelle) gilt also, daß die 2. Ableitung eine Nullstelle haben muß. Weiterhin wurde festgestellt, daß die 2. Ableitung aus dem negativen in den positiven

Bereich übergeht, sie ist also steigend. Dies ist aber wiederum gleichbedeutend mit der Aussage, daß die Steigung der 2. Ableitung (= 3. Ableitung) positiv ist.

Zusammenfassend lauten die Bedingungen für eine Rechts/Links-Wendestelle:

R/L-Wendestelle: 2. Ableitung = 0
3. Ableitung > 0

Einen analogen Beweis kann man für eine Links/Rechts-Wendestelle führen:

L/R-Wendestelle: 2. Ableitung = 0
3. Ableitung < 0

Beispiel

Von der Funktion

$$y = x^3 + 3x^2 - 9x + 5$$

sollen Extremwerte und Wendestellen bestimmt werden. Um diese Aufgabe lösen zu können, werden zunächst die ersten drei Ableitungen der Funktion ermittelt (vgl. Abschnitt 2.1)

$$y' = 3x^2 + 6x - 9$$

$$y'' = 6x + 6$$

$$y''' = 6$$

Extremwerte

Bedingung für die Extremwerte war: $y' = 0$. Also werden die Nullstellen der 1. Ableitung gesucht (vgl. Abschnitt 1.7):

$$0 = 3x^2 + 6x - 9$$

Normalform: $$0 = x^2 + 2x - 3$$

$$x = -\frac{2}{2} + \sqrt{\left(\frac{2}{2}\right)^2 + 3} = 1$$

$$x = -\frac{2}{2} - \sqrt{\left(\frac{2}{2}\right)^2 + 3} = -3$$

Bei $x = 1$ und $x = -3$ liegen also Extremstellen vor. Um zu prüfen, ob ein Minimum oder ein Maximum vorhanden ist, werden für diese x-Werte die jeweiligen Werte der 2. Ableitung berechnet.

$$x = 1 \qquad y'' = 12$$

$$x = -3 \qquad y'' = -12$$

für $x = 1$ ist $y' = 0$ und $y'' > 0$; somit liegt ein Minimum vor. Für $x = -3$ dagegen ist $y' = 0$ und $y'' < 0$, womit nachgewiesen ist, daß es sich an dieser Stelle um ein Minimum der Funktion handelt.

Wendestellen

Wenn eine Wendestelle vorliegt, muß die 2. Ableitung zu 0 werden.

$$0 = 6x + 6$$

$$x = -1$$

Bei $x = -1$ hat die Funktion also eine Wendestelle. Für die Entscheidung Links/Rechts- oder Rechts/Links-Wendestelle wird der Wert der 3. Ableitung für $x = -1$ untersucht.

$$x = -1 \qquad y''' = 6$$

Da y''' größer als 0 ist, handelt es sich um eine Rechts/Links-Wendestelle.

In Bild 2.6 sind die Funktion und alle drei Ableitungen dargestellt, um die in diesem Abschnitt angesprochenen Zusammenhänge zu verdeutlichen.

Allgemeine Literaturhinweise zu Kapitel 2

G, W: [4]; G, W: [6], S. 166 ff.; G, W: [22], S. 252 ff.; G, W: [24], S. 98 ff.; W: [20]

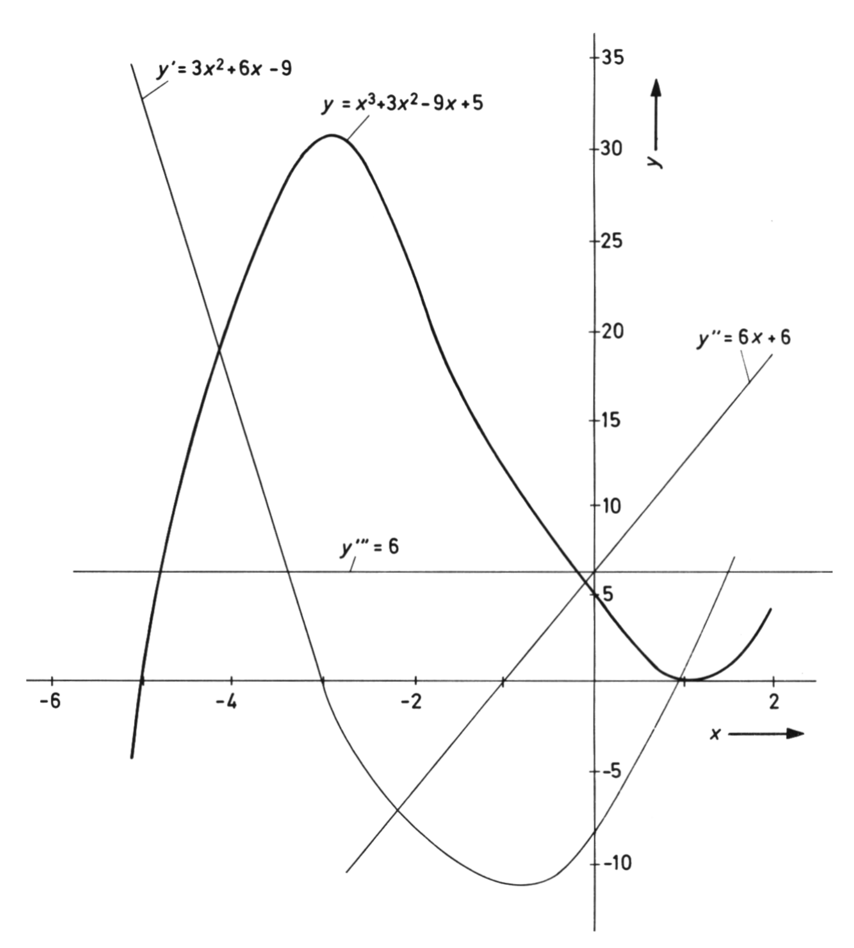


Bild 2.6 Darstellung der Funktion $y = x^3 + 3x^2 - 9x + 5$ mit ihren Ableitungen

Programmbeschreibung

Mit Hilfe des Programms Extremwerte können alle Nullstellen, alle Minima/Maxima und alle Wendestellen einer Funktion bestimmt werden; hierzu werden die Nullstellen der Funktion bzw. der entsprechenden Ableitungen mit Hilfe des in Abschnitt 1.6 beschriebenen Programms Nullrechner ermittelt. Demzufolge müssen die Funktionsgleichung und die ersten drei Ableitungen dem Programm vorliegen; sie sind in den

Zeilen 40	
bis 150	abgelegt. Die
Zeilen 160	
bis 270	stellen das Hauptprogramm dar, wobei in den
Zeilen 170	
bis 210	die einzelnen zur Berechnung notwendigen Unterprogramme aufgerufen werden. In den
Zeilen 220	
bis 270	ist die Abfrage abgelegt, ob eine weitere Funktion untersucht werden soll. In den
Zeilen 670	
bis 870	wird vom Rechner auf die Notwendigkeit hingewiesen, daß alle drei Ableitungen dem Programm vorliegen müssen. Falls das noch nicht der Fall ist, hat man hier die Möglichkeit, sie noch einzugeben. Das Programm stoppt in diesem Fall; nach Eingabe der Gleichungen wird es mit GOTO 860 wieder gestartet. Die Eingabe von Untersuchungsgrenzen und Genauigkeit der Nullstellenbestimmungen erfolgt in den
Zeilen 280	
bis 350	Die Anzahl der Rechenschritte wird in
Zeile 320	hierbei auf 100 festgelegt (vgl. Abschnitt 1.6). Die
Zeilen 360	
bis 440	stellen die Benutzerführung vor der Nullstellenbestimmung dar; die eigenliche Rechnung findet in den
Zeilen 880	
bis 1130	statt. Dieser Programmabschnitt entspricht dem Programm Nullrechner; lediglich die Berechnung des Funktionswertes

mußte etwas komplizierter gestaltet werden, da das gleiche Unterprogramm auch zur Bestimmung der Nullstellen der Ableitungen benutzt wird; somit sind die Sprungadresse für das GOSUB aber variabel.

Hier wurde das Problem so gelöst, daß zunächst die Variable *wo* in

Zeile 410 auf 1 gesetzt wird und hiermit dann in
Zeile 940 sowie in
Zeile 1000
Zeile 1070 und
Zeile 1090 die richtige Sprungadresse für die Funktionsgleichung ausgesucht wird.

Wenn die Rechnung zu den Extremwerten kommt, wird *wo* in

Zeile 510 auf 2 gesetzt und dann in den gleichen Zeilen die Gleichung der 1. Ableitung zur Nullstellenbestimmung benutzt.

Das Entsprechende geschieht für die Wendestellen; hier wird *wo* in

Zeile 610 auf 3 gesetzt und damit dann für die 3. Ableitung die Nullstellen berechnet.

Ist eine Nullstelle gefunden, wird sie ausgegeben und die Rechnung fortgesetzt. Wenn die Intervallgrenze erreicht ist, erfolgt der Sprung zurück ins Hauptprogramm, wo dann mit dem nächsten Programmabschnitt weitergemacht wird. Wenn die Extremstellen berechnet werden, wird neben den x-Werten auch noch der Wert der 1. Ableitung über

Zeile 1150 für diesen x-Wert ausgegeben; dieser muß gleich 0 sein, wenn tatsächlich eine Nullstelle vorliegt. In

Zeile 1160 wird außerdem noch der Wert der 2. Ableitung ausgegeben, damit man entscheiden kann, ob ein Maximum oder ein Minimum vorliegt. Die Berechnung der Wendestellen erfolgt analog zu der Bestimmung der Extremwerte.

Bei der Ausgabe der jeweils berechneten Werte wartet das Programm so lange, bis eine beliebige Taste gedrückt wird. Dies wird erreicht durch die INKEY-Funktion in

Zeile 1120 Vorher wird in
Zeile 1110 der entsprechende Text ausgegeben.

Genau der gleiche Effekt wird erreicht durch Aufrufen einer

entsprechenden Maschinenspracheroutine; dies geschieht mit CALL &BB06.
Dieser Befehl ist zwar etwas kürzer zu tippen; die Adresse stimmt jedoch nur für den CPC 464. Um eine möglichst einfache Übertragung der Programme zu gewährleisten, wird in diesem Buch nur mit der INKEY-Funktion gearbeitet.

Programmlänge: etwa 4,6 K.

```
10 REM Programm extremwerte
20 REM Mit diesem Programm werden die Extremwerte vo
n beliebigen Funktionen berechnet (einschliesslich N
ullstellen)
30 GOTO 160
40 REM Funktionsgleichung
50 y = x*x*x + 3*x*x - 9*x + 5
60 RETURN
70 REM 1.Ableitung
80 y= 3*x*x + 6*x - 9
90 RETURN
100 REM 2.Ableitung
110 y= 6*x + 6
120 RETURN
130  REM 3.Ableitung
140 y=6
150 RETURN
160 REM hier geht's los
170 GOTO 670 : REM Eingabe der Funktion und der Able
itungen
180 GOSUB 280 : REM Untersuchungsgrenzen und Genauig
keit
```

```
190 GOSUB 360 : REM Nullstellen
200 GOSUB 470 : REM Extremstellen
210 GOSUB 570 : REM Wendestellen
220 CLS : PRINT"Sollen von einer weiteren Funktion d
ie Extremwerte bestimmt werden ( /n) ";:INPUT a$
230 IF a$="" THEN RUN
240 IF a$<>"n" THEN 220
250 CLS : LOCATE 20,10 : PRINT"Na, denn man Tschuess
"
260 LOCATE 1,24
270 END
280 REM UP Eingabe von Untersuchungsgrenzen und Gena
uigkeit
290  CLS : PRINT"Geben Sie das Intervall fuer die Un
tersuchungen ein."
300 PRINT:PRINT TAB(20);"xmin ";:INPUT xs
310 PRINT:PRINT TAB(20);"xmax ";:INPUT xe
320 sw0 = (xe-xs)/100 : REM Schrittweite
330  PRINT:PRINT"Mit welcher Genauigkeit sollen die
Extremwerte berechnet werden ?"
340 PRINT:PRINT"Geben Sie die maximale Abweichung ei
n. ";:INPUT abw
350 RETURN
360 REM UP Nullstellenbestimmung
370 CLS : INPUT"Sollen die Nullstellen der Funktion
bestimmt werden ( /n)";a$
380 IF a$ = "" THEN 400
390 IF a$ = "n" THEN 460 ELSE 360
400 b$ = "Nullstelle" : c$ = "y ="
410 wo = 1
420 CLS : PRINT"Nullstellen der Funktion"
```

```
430 PRINT"______________________________________________
_____________________________________"
440 PRINT
450 GOSUB 880 : REM Nullstellenberechnung
460 RETURN
470 REM UP Extremwertbestimmung
480 CLS : PRINT"Sollen die Extremwerte der Funktion
bestimmt werden ( /n) ";:INPUT a$
490 IF a$="" THEN 510
500 IF a$="n" THEN 560 ELSE 470
510 b$="Extremwert" : c$ = "y' =" : wo = 2
520 CLS : PRINT"Extremwerte der Funktion"
530 PRINT"______________________________________________
_____________________________________"
540 PRINT
550 GOSUB 880 : REM Extremwertberechnung
560 RETURN
570 REM UP Wendestellenbestimmung
580 CLS : PRINT"Sollen die Wendestellen der Funktion
 bestimmt werden ( /n) ";:INPUT a$
590 IF a$="" THEN 610
600 IF a$="n" THEN 660 ELSE 570
610 b$="Wendestellen" : c$ = "y'' =" : wo = 3
620 CLS : PRINT"Wendestellen der Funktion"
630 PRINT"______________________________________________
_____________________________________"
640 PRINT
650 GOSUB 880 : REM Wendestellenbestimmung
660 RETURN
670 REM Funktionseingabe fuer beliebige Funktionen
680 CLS
```

```
690 PRINT"Funktionsgleichung und je nach Anwendungsf
all 1.-3.Ableitung muessen dem       Programm vorli
egen, und zwar die"
700 PRINT:PRINT TAB(20);"- Funktionsgleichung in Zei
le 50"
710 PRINT:PRINT TAB(20);"- 1. Ableitung in Zeile 80"
720 PRINT:PRINT TAB(20);"- 2. Ableitung in Zeile 110
"
730 PRINT:PRINT TAB(20);"- 3. Ableitung in Zeile 140
"
740 PRINT"____________________________________________
____________________________________"
750 PRINT:PRINT TAB(20);"Das ist der Fall";TAB(60);"
(1)"
760 PRINT:PRINT TAB(20);"Das muss noch gemacht werde
n";TAB(60);"(2)"
770 INPUT a
780 IF a<1 OR a>2 THEN 670
790 IF a=1 THEN 870
800 LOCATE 1,14 : PRINT "
"
810 LOCATE 1,16 : PRINT"
                                                    "
820 PRINT"                    "
830 LOCATE 1,14
840 PRINT:PRINT"Geben Sie die Gleichungen jeweils in
 der Form  y=.... in den oben angegeben     Zeilen e
in. Starten Sie das Programm
 anschliessend wieder mit  goto 860"
850 STOP
860 REM weiter geht's
```

```
870 GOTO 180
880 REM Programm NullRechner
890 REM Die Nullstelle wird berechnet, indem die Kur
ve mit vorgegebener Schritt- weite abgetastet wird,
bei Vorzeichenwechsel die Sc
hrittweite halbiert wird etc.
900 ON ERROR GOTO 910
910 RESUME NEXT
920 Zaehler = 0
930 x=xs
940 ON wo GOSUB 50,80,110
950 y1=y
960 k=2 : sw = sw0
970 Zaehler = Zaehler + 1
980  x=x+sw
990 IF x>= xe THEN 1110
1000 ON wo GOSUB 50,80,110
1010 y2=y : IF ABS(y) < ABS(abw) THEN 1050
1020 IF SGN(y1)=SGN(y2) THEN 1030 ELSE 1040

1030 y1=y2 : GOTO 980
1040 sw=-sw/k : y1=y2 : GOTO 980
1050 PRINT:PRINT b$;Zaehler;" = ";x,
1060 GOSUB 50 : PRINT"y = ";y
1070 ON wo GOSUB 1130,1140,1180
1080 x=x+sw0
1090 ON wo GOSUB 50,80,110
1100 sw = sw0 : y1=y : GOTO 970
1110 PRINT:PRINT"Weiter mit beliebiger Taste"
1120 IF INKEY$="" THEN 1120
1130 RETURN
```

```
1140 REM UP Ausgabeprogramm fuer Extremwertsuche
1150 GOSUB 80 : PRINT TAB(12);"y' = ";y,
1160 GOSUB 110 : PRINT"y'' = ";y
1170 RETURN
1180 REM UP Ausgabeproramm fuer Wendestellensuche
1190 GOSUB 110 : PRINT TAB(13);"y'' = ";y,
1200 GOSUB 140 : PRINT"y''' = ";y
1210 RETURN
```

3
Integralrechnung

3.1 Integration nach der Trapezregel

Grundlagen

Die Bestimmung eines Flächeninhaltes ist nur lösbar, wenn die Fläche durch Geraden begrenzt wird bzw. wenn geometrisch genau definierte Formen wie Kreise, Elipsen usw. vorliegen. Schwieriger wird es bei der in Bild 3.1 dargestellten Figur:

Es handelt sich hier um die in der Technik sehr häufig vorkommende Aufgabe, die Fläche zwischen Kurve und x-Achse innerhalb der Grenzen x_A und x_E zu bestimmen. Diese Fläche wird auch als bestimmtes Integral bezeichnet. Ist die Funktion der Kurve bekannt, wird man normalerweise in der Lage sein, die Kurve zu integrieren und somit die Fläche zu berechnen (vgl. z.B. [6], S. 261 ff).

Es besteht jedoch auch die Möglichkeit, numerisch zu integrieren, d.h. Näherungsverfahren anzuwenden. Solche Verfahren muß man auch dann benutzen, wenn die Funktion nicht integrierbar ist.

Will man die Integration nach der sogenannten Trapezregel vornehmen, teilt man die zu berechnende Fläche in n gleich breite Streifen.

Die Breite eines solchen Streifens ist

$$h = \frac{x_a - x_e}{n} \qquad (3.1.1)$$

Verbindet man die Funktionswerte an den einzelnen Stellen x_i durch Geraden, so berechnet sich die Fläche eines einzelnen Streifens zu

$$A = \frac{y_{i+1} + y_i}{2} \cdot h \qquad (3.1.2)$$

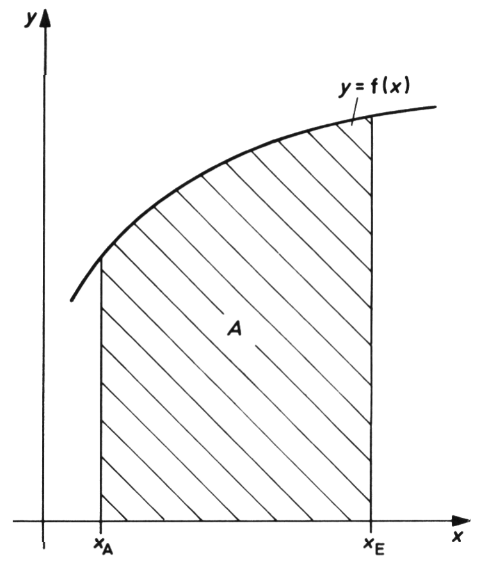


Bild 3.1 Fläche unter einer Kurve

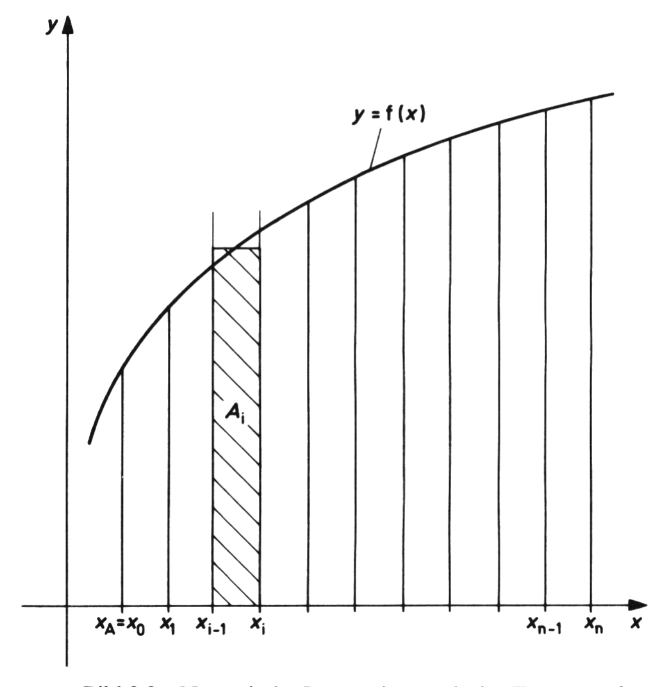


Bild 3.2 Numerische Integration nach der Trapezregel

Die Summe aller einzelnen Streifen ergibt aber die gesamte Fläche:

$$A = \sum_{i=1}^{n} A_i$$

$$A = h\left(\frac{y_0 + y_1}{2} + \frac{y_1 + y_2}{2} + \ldots + \frac{y_{n-1} + y_n}{2}\right)$$

$$A = h\left(\frac{y_0}{2} + y_1 + y_2 + \ldots + y_{n-1} + \frac{y_n}{2}\right) \tag{3.1.3}$$

Gleichung (3.1.3) stellt die gesuchte Trapezregel dar. Je höher man die Anzahl n der Streifen wählt, desto genauer wird die Rechnung, da dann durch die schmäler werdenden Streifen die Abweichungen vom tatsächlichen Verlauf der Funktion abnehmen.

Beispiel

Die Fläche zwischen der Funktion

$$y = 2/x$$

und der x-Achse soll in den Grenzen $x_A = 2$ und $x_E = 4$ bestimmt werden. Als Streifenanzahl wird $n = 10$ gewählt.

Zur Lösung stellt man folgende Tabelle auf:

1	x_i	$y_i = f(x_i) = 2/x_i$
0	2,0	1
1	2,2	0,909090909
2	2,4	0,833333333
3	2,6	0,769230769
4	2,8	0,714285714
5	3,0	0,666666667
6	3,2	0,625
7	3,4	0,588235294
8	3,6	0,555555556
9	3,8	0,526315789
10	4,0	0,5

Als Streifenbreite *h ergibt sich*

$$h = \frac{4-2}{10} = 0{,}2$$

Somit berechnet sich die Fläche zu

$$A = h(y_0/2 + \sum_{i=1}^{q} y_i + y_{10}/2)$$

$$= 1{,}38754281$$

Die Fläche unter der Kurve beträgt also 1,39 Flächeneinheiten.

Literaturhinweise

G: [4], S. 304; G: [5], S. 356 bis 358; G: [22], S. 399 bis 400

Programmbeschreibung

	Die zu untersuchende Funktionsgleichung ist wieder in
Zeile 50	abgelegt. Über
Zeile 90	bzw. das Unterprogramm in den
Zeilen 1000	bis
bis 1050	erfolgt die Eingabe des Intervalles, in dem die Funktion integriert werden soll sowie die Vorgabe der Schrittweite.
	Die Berechnung des Integrals wird in
Zeile 110	bzw. in
Zeile 2100	
bis 2220	vorgenommen, wobei die Sehnen-Trapez-Formel zum Tragen kommt.
	Das Ergebnis wird als Variable *integ* gespeichert. In
Zeile 130	erfolgt über das Unterprogramm in den
Zeilen 1300	
bis 1390	die Ausgabe des Integrals sowie der Intervallgrenzen.
	Länge des Programms: etwa 1,3 K.

```
10 REM Programm Itrapez
20 REM Integration einer vorgegebenen Funktion mit d
er Sehnen-Trapez-Formel
30 GOTO 70
40 REM Zeile 50 = Funktionsgleichung
50 y=1/x
60 RETURN
70 REM los geht's
80 MODE 2
90 GOSUB 1000 : REM Startwerteingabe etc.
100 CLS
110 LOCATE 20,10 : PRINT"Ich arbeite. Bitte warten."
120 GOSUB 2100 : REM Integration nach der Sehnen-Tra
pez-Formel
130 GOSUB 1300 : REM Ausgabe
140 END
1000 REM UP Startwerteingabe etc.
1010 CLS
1020 PRINT"In welchen Grenzen soll die Funktion inte
griert werden ?"
1030 PRINT:PRINT"von xmin = ";:INPUT xmin:PRINT"bis
xmax = ";:INPUT xmax
1040 PRINT:PRINT"Mit welcher Schrittweite soll der R
echner vorgehen ";:INPUT h
1050 RETURN
1300 REM UP ausgabe
1310 CLS
1320 PRINT a$ : PRINT"_________________________" : P
RINT
1330 PRINT"Integral : ";integ
1340 PRINT:PRINT"xmin : ";xmin
```

```
1350 PRINT:PRINT"xmax : ";xmax
1360 LOCATE 1,15 : PRINT"Noch eine Berechnung ( /n)"
;:INPUT a$
1370 IF a$="" THEN RUN
1380 IF a$="n" THEN 1390 ELSE 1360
1390 RETURN
2100 REM UP Integration nach der Sehnen-Trapez-Forme
l
2110 a$="Sehnen-Trapez-Formel"
2120 integ=0 : x=xmin
2130 GOSUB 50 : REM Funktionsgleichung
2140 integ = integ + y/2
2150 x=xmax : GOSUB 50
2160 integ = integ + y/2
2170 FOR x=xmin+h TO xmax-h/2 STEP h
2180 GOSUB 50
2190 integ = integ +y
2200 NEXT x
2210 integ = integ*h
2220 RETURN
```

3.2 Integration nach der Simpsonschen Regel

Grundlagen

Wie auch die Trapezregel zählt man die Simpsonsche Regel zu den Näherungsverfahren, mit deren Hilfe man Funktionen numerisch integrieren kann; gegenüber dieser besteht der Unterschied, daß die y-Punkte nicht durch eine Gerade (s. Bild 3.2), sondern durch eine Parabel verbunden werden. Außerdem muß die Anzahl der Streifen gerade sein.

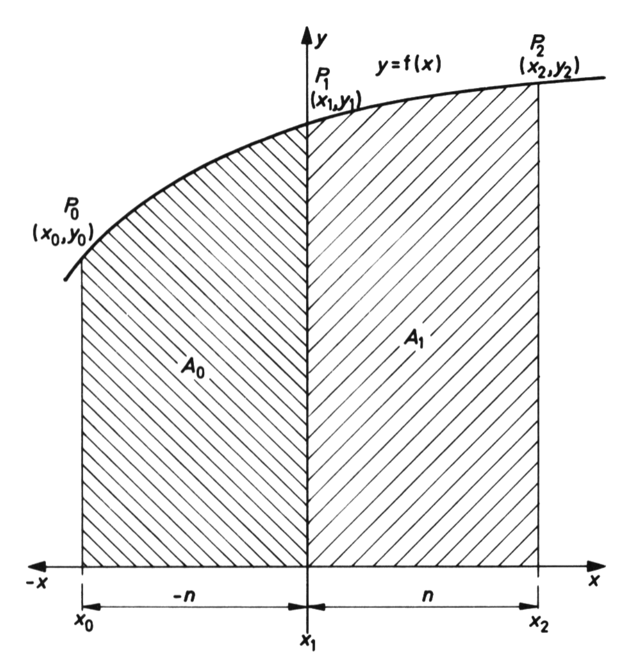


Bild 3.3 Integration nach der Simpsonschen Regel

Will man die Simpsonsche Formel anwenden, faßt man die Streifen aus Bild 3.2 zu Doppelstreifen zusammen. Des weiteren verschiebt man das Koordinatensystem so, daß die y-Achse durch den Punkt P_1 geht.

Die gesamte Fläche $A_0 + A_1$ wird als A_{02} bezeichnet, d.h., die Fläche zwischen den Punkten x_0, y_0, x_2 und y_2.

Nach Bild 3.3 kann man für die drei Punkte P_0, P_1 und P_2 durch die spezielle Wahl des Koordinatensystems auch schreiben:

$$P_0 = (x_0, y_0) = (-h, y_0)$$
$$P_1 = (x_1, y_1) = \quad (0, y_1)$$
$$P_2 = (x_2, y_2) = (+h, y_2)$$

Gesucht wird jetzt die Parabel, die durch alle drei Punkte geht. Hierzu werden folgende Gleichungen aufgestellt:

$$y \ = a_2 x^2 + a_1 x + a_0 \quad \text{(allgemeine Gleichung)} \tag{3.2.1}$$

$$y_0 = a_2 h^2 - a_1 h + a_0 \quad (x \mathrel{\widehat{=}} -h) \tag{3.2.2}$$

$$y_1 = a_0 \quad (x = 0) \tag{3.2.3}$$

$$y_2 = a_2 h^2 + a_1 h + a_0 \quad (x \mathrel{\widehat{=}} h) \tag{3.2.4}$$

Aus Gleichung 3.2.3 folgt direkt:

$$a_0 = y_1 \tag{3.2.5}$$

Addition von Gleichung (3.2.2) und Gleichung (3.2.4) ergibt a_2:

$$y_0 + y_2 = a_2 h^2 - a_1 h + y_1 + a_2 h^2 + a_1 h + a_0$$

$$y_0 + y_2 = 2a_2 h^2 + 2y_1$$

$$a_2 = \frac{y_0 + y_2 - 2y_1}{2h^2} \tag{3.2.6}$$

Subtraktion von Gleichung 3.2.2 von Gleichung 3.2.4 ergibt a_1:

$$y_2 - y_0 = a_2 h^2 + a_1 h + y_1 - a_2 h^2 + a_1 h - y_1$$

$$y_2 - y_0 = 2a_1 h$$

$$a_1 = \frac{y_2 - y_0}{2h} \tag{3.2.7}$$

A_{02} ist aber wiederum nichts anderes als das bestimmte Integral der Parabel in Gleichung (3.2.1) in den Grenzen $-h$ und h.

(Für die folgenden Rechenschritte bzw. Schreibweisen vgl. z.B. [22], S. 357 ff)

$$A_{02} = \int_{-h}^{h} (a_2 x^2 + a_1 x + a_0)\, dx$$

$$= \left[{}^1\!/_3 a_2 x^3 + \frac{a_1}{2} x^2 + a_0 x \right]_{-h}^{h}$$

$$= \frac{a_2 h^3}{3} + \frac{a_1 h^2}{2} + a_0 h + \frac{a_2 h^3}{3} - \frac{a_1 h^2}{2} + a_0 h$$

$$= 2\,\frac{a_2 h^3}{3} + 2a_0 h$$

$$A_{02} = \frac{2(y_0 - 2y_1 + y_2)h^3}{2 \cdot 3h^2} + 2y_1 h$$

$$= \frac{(y_0 - 2y_1 + y_2)h}{3} + 2y_1 h$$

$$= \frac{h}{3}(y_0 + 4y_1 + y_2)$$

Führt man die gleiche Rechnung für die angrenzende Fläche A_{24} durch (das Koordinatensystem legt man durch x_3), so erhält man die gleiche Formel, jedoch sind die Indizes von y jeweils um 2 erhöht.

$$A_{24} = \frac{h}{3}(y_2 + 4y_3 + y_4)$$

Das Gleiche gilt für die dann angrenzenden Flächen:

$$A_{46} = \frac{h}{3}(y_4 + 4y_5 + y_6) \qquad \text{usw.}$$

Die Gesamtfläche unter der Kurve (s. Bild 3.1) errechnet sich als Summe der Einzelflächen:

$$A = A_{02} + A_{24} + A_{46} + \ldots + A_{n-2,n}$$

Setzt man für die einzelnen Flächen die entsprechenden Gleichungen ein, so erhält man:

$$A = \frac{h}{3}[(y_0 + 4y_1 + y_2) + (y_2 + 4y_3 + y_4) + \ldots$$
$$+ (y_{n-2} + 4y_{n-1} + y_n)]$$

Diese Gleichung liefert nach weiteren Umformungen die Simpsonsche Formel:

$$A = \frac{h}{3}(y_0 + 4y_1 + 2y_2 + 4y_3 + 2y_4 + \ldots + 2y_{n-2} + 4y_{n-1} + y_n) \tag{3.2.8}$$

$$A = \frac{h}{3}[y_0 + y_n + 4(y_1 + y_3 + \ldots y_{n-1}) + 2(y_2 + y_4 + \ldots + + y_{n-2})]$$

Beispiel

Um den Unterschied zu verdeutlichen, der zwischen der Trapezregel und der Simpsonschen Regel besteht, soll die gleiche Aufgabe wie in Abschnitt 3.1 gelöst werden. Es geht also um die Integration der Funktion

$$y = 2/x$$

in den Grenzen $x = 2$ und $x = 4$. Als Anzahl der Streifen wird wieder 10 gewählt; die Streifenbreite ist somit wieder 0,2.

Der genaue Wert der Fläche ist 1,38629436 Flächeneinheiten (aus einer geschlossenen Integration berechnet). Zur Anwendung der Simpsonschen Regel stellt man zweckmäßigerweise folgende Tabelle auf:

1	x	$y_i = f(x_i) = 2/x$		
0	2,0	1		
1	2,2		0,909090909	
2	2,4			0,833333333
3	2,6		0,769230769	
4	2,8			0,714285714
5	3,0		0,666666667	
6	3,2			0,625
7	3,4		0,588235294	
8	3,6			0,555555556
9	3,8		0,526315789	
10	4,0	0,5		
		1,5 $=\sum a$	3,459539428 $=\sum b$	2,728174603 $=\sum c$

Die Fläche A ist laut Gleichung (3.2.8):

$$A = \frac{h}{3}\left(\sum a + 4\sum b + 2\sum c\right)$$

$$A = \frac{0,2}{3}(1,5 + 4 * 3,459539428 + 2 * 2,728174603)$$

$$A = 1,38630046 \text{ Flächeneinheiten}$$

Trapezregel:

$$A = 1{,}38754281 \text{ Flächeneinheiten}$$

Genauer Wert:

$$A = 1{,}38629436 \text{ Flächeneinheiten}$$

In diesem Fall ist die Simpsonsche Regel also zwei Dezimalstellen genauer als die Trapezregel. Diese Aussage kann man verallgemeinern; die Simpsonregel ist immer genauer (bzw. in Grenzfällen gleich genau) als die Trapezregel. Die Ursache hierfür ist die bessere Anpassungsmöglichkeit an die integrierende Funktion durch die Parabel.

Literaturhinweise

G: [4], S. 304 bis 305; G: [5], S. 356 bis 360; G: [22], S. 399 bis 402

Programmbeschreibung

Das Programm Isimpson ist genauso aufgebaut wie das in Abschnitt 3.1 beschriebene Programm Itrapez; selbst die Zeilennummern sind gleich. Aus diesem Grund sei hier auf eine detaillierte Programmbeschreibung verzichtet. Der einzige Unterschied besteht darin, daß im Unterprogramm nach Zeile 1000 nicht die Sehnen-Trapez-Formel, sondern die Simpsonsche Regel angewendet wird.

Programmlänge: etwa 1,4 K.

```
10 REM Programm Isimpson
20 REM Integration einer vorgegebenen Funktion mit d
er Simpson'schen Formel
30 GOTO 70
40 REM Zeile 50 = Funktionsgleichung
50 y=SQR(1+EXP(2*x))
60 RETURN
70 REM los geht's
80 MODE 2
```

```
90 GOSUB 1000 : REM Startwerteingabe etc.
100 CLS
110 LOCATE 20,10 : PRINT"Ich arbeite. Bitte warten."
120 GOSUB 1100 : REM Berechnung nach Simpson
130 GOSUB 1300 : REM Ausgabe
140 END
1000 REM UP Startwerteingabe etc.
1010 CLS
1020 PRINT"In welchen Grenzen soll die Funktion inte
griert werden ?"
1030 PRINT:PRINT"von xmin = ";:INPUT xmin:PRINT"bis
xmax = ";:INPUT xmax
1040 PRINT:PRINT"Mit welcher Schrittweite soll der R
echner vorgehen ";:INPUT h
1050 RETURN
1100 REM UP Simpson-Berechnung
1110 a$="Simpson'sche Formel"
1120 integ = 0 : x=xmin
1130 GOSUB 50 : REM Funktion
1140 integ = integ + y
1150 x=xmax : GOSUB 50
1160 integ = integ + y
1170 FOR x=xmin+h TO xmax STEP h*2
1180 GOSUB 50
1190 integ = integ + y*4
1200 NEXT x
1210 FOR x=xmin+2*h TO xmax-h STEP 2*h
1220 GOSUB 50
1230 integ = integ + 2*y
1240 NEXT x
1250 integ = integ*h/3
```

```
1260 RETURN
1300 REM UP ausgabe
1310 CLS
1320 PRINT a$ : PRINT"__________________________" : P
RINT
1330 PRINT"Integral : ";integ
1340 PRINT:PRINT"xmin : ";xmin
1350 PRINT:PRINT"xmax : ";xmax
1360 LOCATE 1,15 : PRINT"Noch eine Berechnung ( /n)"
;:INPUT a$
1370 IF a$="" THEN RUN
1380 IF a$="n" THEN 1390 ELSE 1360
1390 RETURN
```

4
Vektoralgebra

4.1 Grundrechenarten

Grundlagen

Ein Skalar ist eine Größe, die durch eine einzige Angabe eindeutig beschrieben ist, wie z.B. Masse, Zeit oder auch Länge. Ein Vektor dagegen braucht zur eindeutigen Bestimmung einen Skalar, der den Betrag des Vektors angibt, und eine Richtungsangabe, in die der Vektor wirkt. Beispiele hierfür sind Kräfte, Momente oder Geschwindigkeiten. Nicht genau definiert werden muß der genaue Ort, an dem der Vektor beginnt; man kann Vektoren beliebig parallel verschieben.

In Bild 4.1 sind die beiden Vektoren $\vec{a}$ und $\vec{b}$ in ein Koordinatensystem eingezeichnet. Um sie eindeutig zu bestimmen, gibt es zwei Möglichkeiten: Entweder gibt man den jeweiligen Betrag $a = |\vec{a}|$ bzw. $b = |\vec{b}|$ an sowie den Winkel α_a bzw. α_b, den sie mit der x-Achse einschließen, oder aber man gibt die jeweiligen Beträge in x- und y-Richtung für jeden Vektor an. Bei der zuerst genannten Möglichkeit spricht man von Polarkoordinaten, bei der zweiten von rechtwinkligen Koordinaten. Will man den Vektor $\vec{a}$ z.B. in Polarkoordinaten angeben, so lautet die Angabe:

$$\vec{a} = |\vec{a}|\,(\sin\alpha_a + \cos\alpha_a) \tag{4.1.1}$$

In rechtwinkligen Koordinaten lautet die Angabe dagegen:

$$\vec{a} = a_x\vec{i} + a_y\vec{j} \tag{4.1.2}$$

$a_x\vec{i}$ = Vektorkomponente bezüglich der x-Achse
$a_y\vec{j}$ = Vektorkomponente bezüglich der y-Achse

Polar- und rechtwinklige Koordinaten können ineinander umgerechnet werden:

$$a = |\vec{a}| = \sqrt{a_x^2 + a_y^2} \qquad (4.1.3)$$

$$\alpha_a = \arctan\left(\frac{a_y}{a_x}\right) \qquad (4.1.4)$$

$$a_x = |\vec{a}| * \cos\alpha_a \qquad (4.1.5)$$

$$a_y = |\vec{a}| * \sin\alpha_a \qquad (4.1.6)$$

Im Folgenden wird hauptsächlich mit rechtwinkligen Koordinaten gerechnet.

Verschiebt man den Vektor $\vec{b}$ derart, daß sein Beginn an das Ende von Vektor $\vec{a}$ kommt, so hat man die Möglichkeit geschaffen, beide zu addieren. Beim sogenannten Kräfteparallelogramm, kurz Krafteck genannt, zieht man einfach eine Linie vom Beginn des Vektors $\vec{a}$ zum Ende des Vektors $\vec{b}$ und erhält so den resultierenden Vektor $\overrightarrow{a + b}$.

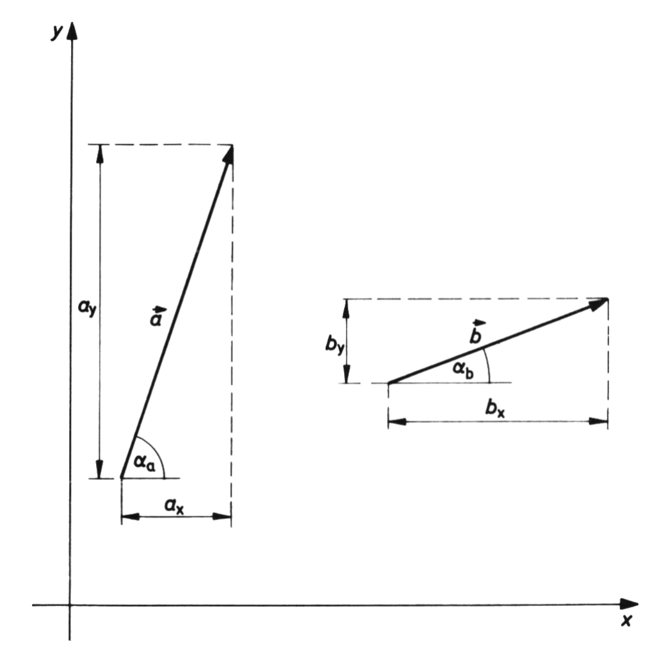


Bild 4.1 Zwei Vektoren in einem Koordinatensystem

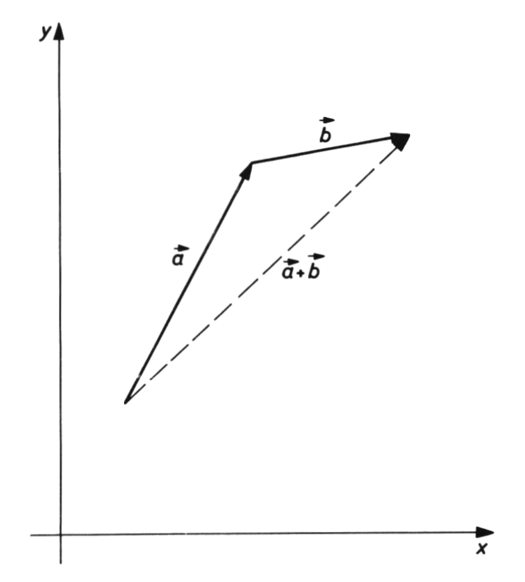


Bild 4.2 Addition zweier Vektoren

Um dies auch rechnerisch zu erfassen, braucht man lediglich die jeweiligen x- und y-Werte beider Vektoren zu addieren, um den resultierenden Vektor zu erhalten.

$$\vec{a} + \vec{b} = (a_x + b_x)\vec{i} + (a_y + b_y)\vec{j} \tag{4.1.7}$$

Will man zwei Vektoren dagegen voneinander subtrahieren, zieht man die beiden x- und y-Koordinaten voneinander ab.

$$\vec{a} - \vec{b} = (a_x - b_x)\vec{i} + (a_y - b_y)\vec{j} \tag{4.1.8}$$

Will man einen Vektor mit einem Skalar, d.h. mit einer Zahl, multiplizieren (z.B.: Die Kraft ist zu verdoppeln), so multipliziert man lediglich den Betrag; der Winkel bleibt erhalten.

$$2\vec{a} = 2\,|\vec{a}|\,(\sin\alpha_a + \cos\alpha_a) \tag{4.1.9}$$

Wenn man in rechtwinkligen Koordinaten rechnet, sind demnach die Beträge in x- und in y-Richtung mit dem Skalar zu multiplizieren:

$$2\vec{a} = 2a_x\vec{i} + 2a_y\vec{j} \tag{4.1.10}$$

Beispiele

1. Der Vektor $\vec{a} = 3\vec{i} + 4\vec{j}$ ist in Polarkoordinaten anzugeben.

$$\begin{aligned} a = |\vec{a}| &= \sqrt{a_x^2 + a_y^2} \\ &= \sqrt{3^2 + 4^2} \\ &= 5 \\ \alpha_a &= \arctan\left(\frac{a_y}{a_x}\right) \\ &= \arctan\left(\frac{4}{3}\right) \\ &= 53{,}1^\circ \end{aligned}$$

2. Die Vektoren $\vec{a} = 3\vec{i} + 4\vec{j}$ sowie $\vec{b} = 5\vec{i} + 10\vec{j}$ sind zu addieren und zu subtrahieren.

$$\begin{aligned} \vec{a} + \vec{b} &= (a_x + b_x)\vec{i} + (a_y + b_y)\vec{j} \\ &= (3 + 5)\vec{i} + (4 + 10)\vec{j} \\ &= 8\vec{i} + 14\vec{j} \end{aligned}$$

$$\begin{aligned} \vec{a} - \vec{b} &= (a_x - b_x)\vec{i} + (a_y - b_y)\vec{j} \\ &= (3 - 5)\vec{i} + (4 - 10)\vec{j} \\ &= -2\vec{i} - 6\vec{j} \end{aligned}$$

3. Der Vektor $\vec{a} = 3\vec{i} + 4\vec{j}$ ist mit 2,5 zu multiplizieren.

$$\begin{aligned} 2{,}5\vec{a} &= 2{,}5a_x\vec{i} + 2{,}5a_y\vec{j} \\ &= 2{,}5 * 3\vec{i} + 2{,}5 * 4\vec{j} \\ &= 7{,}5\vec{i} + 10\vec{j} \end{aligned}$$

Anmerkung

Das Programm zu dem gesamten Kapitel sowie die Literaturhinweise finden sich im Anschluß an Abschnitt 4.4.3.

4.2 Skalarprodukt

Grundlagen

Vektoren lassen sich nicht nur durch die beschriebenen Grundrechenarten miteinander verknüpfen, sondern genügen noch weiteren Rechengesetzen. Eines dieser Gesetze beschreibt die Bildung des sogenannten skalaren Produktes. Das skalare Produkt selbst ist ein Skalar, die Richtungsangabe fällt also weg. Berechnet wird es, indem die jeweiligen Koordinaten der beiden Vektoren multipliziert werden und hiervon die Summe gebildet wird.

$$\vec{a} \cdot \vec{b} = a_x \cdot b_x + a_y \cdot b_y \tag{4.2.1}$$

Kennzeichen für das skalare Produkt ist der Multiplikationspunkt zwischen den beiden Vektoren. Entstanden ist diese Vorschrift, weil eine Beschreibung der Arbeit nötig wurde; diese hängt ab von einer Kraft $\vec{F}$ und dem Weg $\vec{s}$, über den die Kraft insgesamt ansteht.

$$W = \vec{F} \cdot \vec{s} \tag{4.2.2}$$

Die Arbeit ist also das Produkt aus den beiden Vektoren Kraft und Weg; da sie selbst aber unabhängig von einer Richtung ist, ist sie kein Vektor, sondern ein Skalar.

Beispiel

Gegeben ist die Kraft $\vec{F} = F_x\vec{i} + Fy\vec{j}$ mit $F_x = 300\,\text{N}$ und $F_y = 1000\,\text{N}$.

Gesucht ist die Arbeit, die auf dem Weg von $P_1 = (5\,\text{m},\ 2\,\text{m})$ nach $P_2 = (10\,\text{m},\ 1\,\text{m})$ verrichtet wird.

Zunächst wird der Weg $\vec{s}$ bestimmt:

$$\begin{aligned} \vec{s} &= \overrightarrow{P_1P_2} \\ &= (10\,\text{m} - 5\,\text{m})\,\vec{i} + (1\,\text{m} - 2\,\text{m})\,\vec{j} \\ &= 5\,\text{m}\,\vec{i} - 1\,\text{m}\,\vec{j} \end{aligned}$$

Somit errechnet sich die Arbeit wie folgt:

$$\begin{aligned} W &= \vec{F} \cdot \vec{s} \\ &= F_x \cdot s_x + F_y \cdot s_y \\ &= 300\,\mathrm{N} \cdot 5\,\mathrm{m} - 1000\,\mathrm{N} \cdot 1\,\mathrm{m} \\ &= 500\,\mathrm{Nm} \end{aligned}$$

4.3 Vektorprodukt

Grundlagen

Das Vektorprodukt der beiden Vektoren $\vec{a}$ und $\vec{b}$ beschreibt die Fläche des Parallelogramms, das zwischen den beiden Vektoren aufgespannt ist (Bild 4.3).

Kennzeichen der Schreibweise des Vektorproduktes ist das Kreuz zwischen zwei Vektoren. Das Ergebnis ist ein neuer Vektor.

Der Betrag von $\vec{A}$ ist der Flächeninhalt des Parallelogramms. Weiterhin ist festgelegt, daß die Richtung von $\vec{A}$ senkrecht ist zu der Richtung von $\vec{a}$ und der Richtung von $\vec{b}$; $\vec{A}$ steht also senkrecht auf dem Parallelogramm.

Zur Berechnung des Vektorproduktes wird angenommen, daß der Vektor in einem dreidimensionalen Koordinatensystem vorliegt.

$$\vec{a} = a_x \vec{i} + a_y \vec{j} + a_z \vec{k} \tag{4.3.1}$$

$$\vec{b} = b_x \vec{i} + b_y \vec{j} + b_z \vec{k} \tag{4.3.2}$$

$a_x \vec{i}$, $b_x \vec{i}$ = Vektorkomponente in x-Richtung
$a_y \vec{j}$, $b_y \vec{j}$ = dito in y-Richtung
$a_z \vec{k}$, $b_z \vec{k}$ = dito in z-Richtung

Das Vektorprodukt ist jetzt der Wert der folgenden Determinante (zur Berechnung von Determinanten vgl. Kapitel 6).

$$\vec{a} \times \vec{b} = \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{vmatrix} \tag{4.3.3}$$

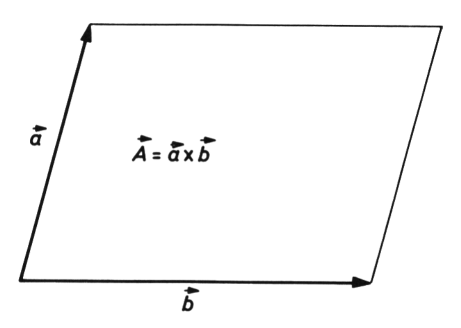


Bild 4.3 Vektorprodukt

Beispiel

Wie groß ist das Parallelogramm zwischen den Vektoren

$$\vec{a} = 3\vec{i} + 2\vec{j} + 0\vec{k} \qquad \text{und} \qquad \vec{b} = 5\vec{i} + 2\vec{j} + 0\vec{k}?$$

Beide Vektoren haben keine z-Komponente, liegen also in einem zweidimensionalen Koordinatensystem. Trotzdem muß nach Gleichung (4.3.3) ein dreidimensionales betrachtet werden.

$$\vec{A} = \vec{a} \cdot \vec{b} = \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ 3 & 2 & 0 \\ 5 & 2 & 0 \end{vmatrix}$$

$$= 0\vec{i} + 0\vec{j} - 4\vec{k}$$

Der Flächenvektor $\vec{A}$ hat also eine alleinige Komponente in z-Richtung, steht somit senkrecht zu $\vec{a}$ und $\vec{b}$.

$$|\vec{A}| = \sqrt{A_x^2 + A_y^2 + A_z^2}$$

(vgl. Abschnitt 4.1)

$$= \sqrt{0^2 + 0^2 + 4^2}$$

$$= 4$$

Die Fläche hat eine Größe von 4 Flächeneinheiten.

Literaturhinweise

G: [19], S. 141 ff; G, W: [1], S. 67 ff; G, W: [5], S. 790 ff; G, W: [7], S. 170 ff; W: [16], S. 9 ff.

Programmbeschreibung

Alle in Kapitel 4 beschriebenen Rechenarten sind im Programm Vektoralgebra zusammengefaßt. Das Programm ist hierbei so aufgebaut, daß auf dem Bildschirm zunächst ein Menü erscheint; hieraus wählt man über eine Code-Zahl die gewünschte Rechenoperation aus. Nach erfolgter Ausführung kehrt das Programm dann jeweils zum Menü zurück.

In
Zeile 30 erfolgt zunächst die Eingabe der Dimension des zu berechnenden Vektorraumes; entsprechend erfolgt in
Zeile 40 die Dimensionierung der Felder v und $v1$, in denen die Vektoren abgespeichert werden.
Zeile 50
bis 180 stellt das erwähnte Menü mit Eingabe der Code-Zahl dar, aufgrund der
Zeile 190 das entsprechende Unterprogramm auswählt.
Zeile 200 bringt das Programm wieder an den Anfang der Menüausgabe zurück, nachdem die Rechnung durchgeführt wurde.

Mittels des Unterprogramms in den
Zeilen 300
bis 460 wird der erste Vektor eingegeben.
Dies geschieht zunächst durch das Unterprogramm in
Zeile 2000
bis 2150 mit dem die Beträge des Vektors in x-, y-, z- usw. Richtung nach $v1$ gespeichert werden. In den
Zeilen 330
bis 350 wird der Vektor dann nach v gespeichert. Dieser erste Programmabschnitt muß in jedem Fall durchlaufen werden, wenn mit Vektoren gerechnet werden soll, da alle weiteren Operationen mit dem in v gespeicherten Vektor ablaufen.

Auch das Ergebnis aller Rechenschritte wird jeweils nach v gespeichert, sofern es einen Vektor darstellt; auf diese Art und Weise sind Kettenrechnungen leicht möglich.
Das etwas umständliche Verfahren der Eingabe wurde gewählt, da auch bei den eigentlichen Rechenoperationen z.T. Vektoren eingegeben werden müssen. Diese können hier über das gleiche Unterprogramm eingegeben werden, lediglich die weitere Behandlung ist dann von Fall zu Fall unterschiedlich.
Dies kann man bereits am Beispiel der

Zeilen 410 bis 460

sehen, die das Unterprogramm für die Addition und die Subtraktion eines Vektors darstellen. Auch hier wird das Unterprogramm in

Zeile 2000 bis 2150

für die Eingabe des neuen Vektors benutzt; dieser wird dann zu dem in v befindlichen Vektor addiert bzw. von diesem abgezogen. Der Ergebnisvektor wird dann wieder nach v gespeichert.
Bei der Faktormultiplikation wird der Faktor in

Zeile 520

eingegeben und jeder Betrag von v gemäß der Rechenregel mit diesem Faktor multipliziert. Auch hier bleibt der Ergebnisvektor in v. Bei der Bildung des Skalarproduktes in

Zeile 600 bis 700

wird zunächst der Multiplikationsvektor eingegeben und dann in den

Zeilen 630 bis 650

das Skalarprodukt gebildet. Dies wird dann sofort ausgegeben, da es keinen Vektor darstellt und somit nicht nach v gespeichert werden kann. Durch Betätigen einer beliebigen Taste springt man dann zum Hauptprogramm zurück. Zur Bildung des Vektorproduktes in

Zeile 800 bis 910

wird zunächst geprüft, ob die betrachteten Vektoren dreidimensional sind, da nur dann das Vektorprodukt errechnet wird. Ist das nicht der Fall, wird in

Zeile 820

der entsprechende Hinweis gegeben und dann zurück zum Hauptmenü gesprungen.

Andernfalls wird der Multiplikationsvektor eingegeben und in den

Zeilen 870
bis 900 das Vektorprodukt berechnet und nach v gespeichert. In den

Zeilen 1000
bis 1370 werden Polar- und rechtwinklige Koordinaten ineinander umgerechnet; nachdem man in

Zeile 1060 eingegeben hat, was gemacht werden soll, fragt das Programm die notwendigen Eingaben ab und gibt die neu berechneten Koordinaten dann aus.

Das vorletzte Unterprogramm in den

Zeilen 1400
bis 1510 dient dazu, den in v befindlichen Vektor auszugeben. Hat man also eine Rechenoperation durchgeführt, deren Ergebnis ein neuer Vektor ist und das nach v gespeichert ist, wird der Vektor nicht automatisch ausgegeben, sondern man muß die Ausgabe erst über das Menü anwählen. Auch dies erscheint im ersten Moment sicherlich etwas umständlich, erleichtert jedoch Kettenrechnungen und dient zudem der Übersichtlichkeit. Mit dem Unterprogramm in den

Zeilen 1600
bis 1690 wird schließlich entsprechend Abschnitt 4.1 der Betrag des in v befindlichen Vektors berechnet.

Das Programm benutzt bei den Berechnungen ausschließlich rechtwinklige Koordinaten, da diese programmtechnisch leichter zu verarbeiten sind. Fehleingaben werden durch die Dialogführung jedoch ausgeschlossen.

Länge des Programms: etwa 5,1 K.

```
10 REM Gesamtprogramm Vektoralgebra
20 MODE 2
30 CLS : INPUT"Wieviele Dimensionen hat der Vektorra
um";dimension
40 DIM v(dimension),v1(dimension)
50 CLS : REM Hauptmenue
60 PRINT"Was soll gemacht werden ?":PRINT"_________
________________________________________________________
_________________"
70 PRINT:PRINT"- der 1.Vektor soll eingegeben werden
";TAB(70);"(1)"
80 PRINT:PRINT"- ein Vektor soll addiert werden";TAB
(70);"(2)"
90 PRINT:PRINT"- ein Vektor soll subtrahiert werden"
;TAB(70);"(3)"
100 PRINT:PRINT"- es soll mit einem Faktor multipliz
iert werden";TAB(70);"(4)"
110 PRINT:PRINT"- es soll ein Skalarprodukt gebildet
 werden";TAB(70);"(5)"
120 PRINT:PRINT"- es soll ein Vektorprodukt gebildet
 werden";TAB(70);"(6)"
130 PRINT:PRINT"- Polar- bzw. rechtwinklige Koordina
ten sollen umgerechnet werden";TAB(70);"(7)"
140 PRINT:PRINT"- der berechnete Vektor soll ausgege
ben werden";TAB(70);"(8)"
150 PRINT:PRINT"- der Betrag des berechneten Vektors
 soll bestimmt werden";TAB(70);"(9)"
160 PRINT:PRINT"- Schluss";TAB(70);"(10)"
170 INPUT was
180 IF was<1 OR was>10 THEN 50
190 ON was GOSUB 300,400,400,500,600,800,1000,1400,1
```

```
600,1700
200 GOTO 50
300 REM Eingabe des 1.Vektors
310 ERASE v : DIM v(dimension)
320 GOSUB 2000 : REM Eingabe nach v1
330 FOR k=1 TO dimension : REM Umspeicherung nach v
340 v(k)=v1(k)
350 NEXT k
360 RETURN
400 REM UP/HP fuer + und -
410 GOSUB 2000 : REM Eingabe des neuen Vektors nach
V1
420 FOR k=1 TO dimension
430 IF was=2 THEN v(k)=v(k) + v1(k)
440 IF was=3 THEN v(k)=v(k) - v1(k)
450 NEXT k
460 RETURN
500 REM UP/HP Faktormultiplikation
510 CLS
520 PRINT"Faktor";:INPUT fak
530 FOR k=1 TO dimension
540 v(k)=v(k) * fak
550 NEXT k
560 RETURN
600 REM HP/UP Skalarprodukt
610 GOSUB 2000 : REM Eingabe des Multiplikationsvekt
ors
620 sp=0
630 FOR k=1 TO dimension
640 sp=sp + v(k)*v1(k)
650 NEXT k
```

```
660 CLS
670 PRINT"Skalarprodukt : ";sp
680 LOCATE 1,23 : PRINT"Weiter mit beliebiger Taste"
690 IF INKEY$="" THEN 690
700 RETURN
800 REM Vektorprodukt
810 IF dimension = 3 THEN 860
820 CLS : PRINT"Vektorprodukte koennen nur von dreid
imensionalen Vektoren berechnet werden      (die z-K
omponente muss u.U. mit 0 ei
ngegeben werden)."
830 LOCATE 1,23 : PRINT"Weiter mit beliebiger Taste"

840 IF INKEY$="" THEN 840
850 RETURN
860 GOSUB 2000 : REM Eingabe des 2.Vektors
870 xb = v(2)*v1(3) - v1(2)*v(3)
880 yb = -v(1)*v1(3) + v1(1)*v(3)
890 zb = v(1)*v1(2) - v1(1)*v(2)
900 v(1)=xb : v(2)=yb : v(3)=zb
910 RETURN
1000 REM UP/HP Umrechnung von Polar- und rechtwinkli
gen Koordinaten
1010 DEG
1020 CLS
1030 PRINT"Was soll in was umgerechnet werden ?"
1040 PRINT:PRINT"- Polarkoordinaten in rechtwinklige
 Koordinaten";TAB(55);"(1)"
1050 PRINT:PRINT"- rechtwinklige Koordinaten in Pola
rkoordinaten";TAB(55);"(2)"
1060 INPUT was
1070 IF was<1 OR was>2 THEN 1000
```

```
1080 IF was=2 THEN GOTO 1250
1090 REM Umrechnung von Polarkoordinaten in rechtwin
klige Koordinaten
1100 CLS : PRINT"Betrag des Vektors : ";:INPUT betra
g
1110 PRINT:INPUT"Winkel zur x-Achse (in Grad) : ";wi
nkel
1120 x=betrag*SIN(winkel)
1130 y=betrag*COS(winkel)
1140 rcos=COS(winkel)
1150 GOSUB 1170 : REM Ausgabe
1160 RETURN
1170 REM Ausgabe-UP
1180 CLS
1190 PRINT"x : ";x,"y : ";y
1200 PRINT:PRINT"Betrag : ";betrag,"Winkel : ";winke
l;"grad","(eventuell nicht richtig, da tan nicht ein
deutig)"
1210 PRINT:PRINT"Richtungscosinus : ";rcos
1220 LOCATE 1,23 : PRINT"Weiter mit beliebiger Taste
"
1230 IF INKEY$="" THEN 1230
1240 RETURN
1250 REM Umrechnung von rechtwinkligen Koordinaten i
n Polarkoordinaten
1260 CLS : PRINT"x-Betrag : ";:INPUT x
1270 PRINT:PRINT"y-Betrag : ";:INPUT y
1280 betrag = SQR(x*x + y*y)
1290 winkel = ATN(y/x)
1300 rcos = x/betrag
1310 GOSUB 1170 : REM Ausgabe
```

```
1320 RETURN
1400 REM Ausgabe des Vektors
1410 CLS
1420 FOR k=1 TO dimension
1430 IF k=1 THEN LOCATE 12,k : PRINT"x : ";
1440 IF k=2 THEN LOCATE 12,k : PRINT"y : ";
1450 IF k=3 THEN LOCATE 12,k : PRINT"z : ";
1460 IF k>3 THEN PRINT"Dimension ";k;": ";
1470 PRINT v(k)
1480 NEXT k
1490 LOCATE 1,23 : PRINT"Weiter mit beliebiger Taste
"
1500 IF INKEY$="" THEN 1500
1510 RETURN
1600 REM Betragsberechnung eines Vektors
1610 sum=0
1620 FOR k=1 TO dimension
1630 sum=sum + v(k)*v(k)
1640 NEXT k
1650 betrag=SQR(sum)
1660 CLS : PRINT"Betrag : ";betrag
1670 LOCATE 1,23 : PRINT"Weiter mit beliebiger Taste
"
1680 IF INKEY$="" THEN 1680
1690 RETURN
1700 REM UP/HP Schluss machen
1710 CLS
1720 LOCATE 20,10 : PRINT"Na gut. Denn man Tschuess.
"
1730 LOCATE 1,23
1740 END
```

```
2000 REM Eingabe des Vektors in das Feld v1
2010 ERASE v1 : DIM v1(dimension)
2020 CLS : PRINT"Geben Sie den Vektor ein. Es werden
 rechtwinklige Koordinaten benoetigt, d.h.,  die Bet
raege des Vektors in x-, y-,
 z-, etc. Richtung":PRINT"_________________________
_____________________________________________________
__"
2030 FOR k=1 TO dimension
2040 LOCATE 1,6
2050 IF k=1 THEN PRINT"x : ";
2060 IF k=2 THEN PRINT"y : ";
2070 IF k=3 THEN PRINT"z : ";
2080 IF k>3 THEN PRINT"Dimension ";k;" : ";
2090 INPUT v1(k)
2100 LOCATE 1,6 : PRINT"
2110 NEXT k
2120 LOCATE 1,23 : PRINT"Eingaben alle i.o. ( /n)";:
 INPUT a$
2130 IF a$="" THEN 2150
2140 IF a$="n" THEN 2000 ELSE 2120
2150 RETURN
```

5
Komplexe Zahlen

Grundlagen

In Naturwissenschaft und Technik ist es bei vielen Gelegenheiten wünschenswert, Gleichungen der Form $x^2 = -1$ zu lösen. Dies ist in der Menge der reellen Zahlen nicht möglich. Um sich aus dieser Verlegenheit zu helfen, hat man die imaginäre Einheit i definiert:

$$\mathrm{i}^2 = -1 \tag{5.1}$$

Durch diese Definition entsteht eine ganze Klasse neuer Zahlen, die komplexen Zahlen. Zum Rechnen mit i verfährt man grundsätzlich wie bei reellen Zahlen; so gilt z.B.

$$\mathrm{i} + \mathrm{i} = 2\mathrm{i} \qquad 3{,}5\mathrm{i} - \mathrm{i} = 2{,}5\mathrm{i}$$

$$\mathrm{i}^0 = 1 \qquad \mathrm{i}^1 = \mathrm{i}$$

$$\mathrm{i}^2 = -1 \qquad \mathrm{i}^3 = \mathrm{i} \cdot \mathrm{i}^2 = -1$$

Hieraus kann man folgende Regeln herleiten:

$$1 = \mathrm{i}^{4n} \quad (5.2) \qquad \mathrm{i} = \mathrm{i}^{4n+1} \quad (5.3)$$

$$-1 = \mathrm{i}^{4n+2} \quad (5.4) \qquad -\mathrm{i} = \mathrm{i}^{4n+3} \quad (5.5)$$

wobei n ein Element der natürlichen Zahlen einschließlich 0 sein kann.

Eine komplexe Zahl Z besteht aus einem reellen und einem sogenannten imaginären Anteil; veranschaulichen kann man sich dies mit der Gaußschen Zahlenebene, wo in der Waagerechten der reelle Teil von Z und in der Senkrechten der imaginäre Teil aufgetragen wird (Bild 5.1).

Definiert ist die komplexe Zahl als Punkt in dieser Ebene:

$$Z = a + b\mathrm{i} \tag{5.6}$$

a = Realteil von Z
$b\mathrm{i}$ = Imaginärteil von Z

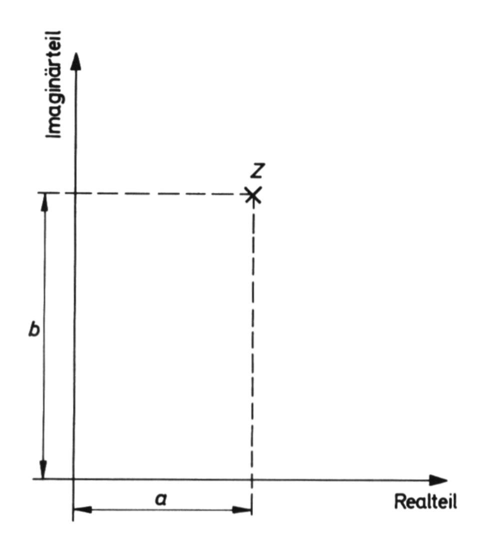


Bild 5.1 Die komplexe Zahl in der Gaußschen Zahlenebene

Verschwindet bi, so hat man eine reelle Zahl vorliegen, die Zahl befindet sich auf der waagerechten Achse. Verschwindet dagegen a, so hat man eine sogenannte imaginäre Zahl; diese liegt auf der senkrechten Achse. Eine «größer als» oder «kleiner als» Beziehung läßt sich bei komplexen Zahlen nicht definieren; dagegen lassen sich die vier Grundrechenarten anwenden. Hierbei wird auch die Verwandschaft mit den in Kapitel 4 besprochenen Vektoren deutlich.

Addition

Zwei komplexe Zahlen werden addiert, indem man jeweils die Realteile und die Imaginärteile addiert.

$$(a_1 + b_1 \mathrm{i}) + (a_2 + b_2 \mathrm{i}) = (a_1 + a_2) + (b_1 + b_2)\mathrm{i} \tag{5.7}$$

Dies entspricht genau der Addition zweier Vektoren; das gilt auch für die

Subtraktion

Zwei komplexe Zahlen werden subtrahiert, indem man jeweils die Realteile und die Imaginärteile voneinander subtrahiert.

$$(a_1 + b_1 \mathrm{i}) - (a_2 + b_2 \mathrm{i}) = (a_1 - a_2) + (b_1 - b_2)\mathrm{i} \tag{5.8}$$

Multiplikation

Bei der Multiplikation zweier komplexer Zahlen werden beide gliedweise miteinander multipliziert.

$$(a_1 + b_1 \mathrm{i}) * (a_2 + b_2 \mathrm{i}) = a_1 a_2 + a_1 b_2 \mathrm{i} + a_2 b_1 \mathrm{i} + b_1 b_2 \mathrm{i}^2$$

Da i^2 aber gleich -1 ist, kann man hierfür auch schreiben:

$$(a_1 + b_1 \mathrm{i}) * (a_2 + b_1 \mathrm{i}) = (a_1 a_2 - b_1 b_2) + (a_1 b_2 + a_2 b_1)\mathrm{i} \tag{5.9}$$

Division

Will man die Zahl $a_1 + b_1 \mathrm{i}$ durch die Zahl $a_2 + b_2 \mathrm{i}$ dividieren, so erweitert man den Bruch zunächst mit dem Konjugierkomplex $a_2 - b_2 \mathrm{i}$

$$\frac{a_1 + b_1 \mathrm{i}}{a_2 + b_2 \mathrm{i}} = \frac{(a_1 + b_1 \mathrm{i})(a_2 - b_2 \mathrm{i})}{(a_2 + b_2 \mathrm{i})(a_2 - b_2 \mathrm{i})}$$

Jetzt wendet man auf Zähler und Nenner jeweils die Multiplikationsregel an:

Zähler: $(a_1 + b_1 \mathrm{i})\,(a_2 - b_2 \mathrm{i}) = (a_1 a_2 + b_1 b_2) - (a_1 b_2 - a_2 b_1)\mathrm{i}$

Nenner: $(a_2 + b_2 \mathrm{i})\,(a_2 - b_2 \mathrm{i}) = (a_2^2 + b_2^2) - (a_2 b_2 - a_2 b_2)\mathrm{i} = a_2^2 + b_2^2$

Somit erhält man als Rechenregel:

$$\frac{a_1 + b_1 \mathrm{i}}{a_2 + b_2 \mathrm{i}} = \frac{a_1 a_2 + b_1 b_2}{a_2^2 + b_2^2} + \frac{-a_1 b_2 + a_2 b_1}{a_2^2 + b_2^2}\,\mathrm{i} \tag{5.10}$$

Komplexe Zahlen kann man genau wie Vektoren auch in einer Polarform angeben. Hierfür dient die sogenannte Zeigerdarstellung (Bild 5.2).

Die Polarform lautet dann:

$$Z = r(\sin\varphi + \mathrm{i}\cos\varphi) \tag{5.11}$$

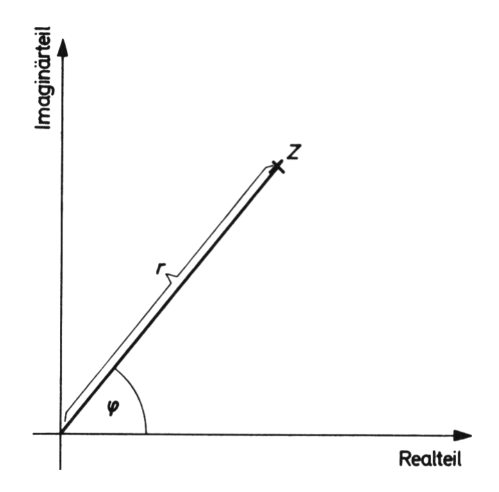


Bild 5.2 Zeigerdarstellung der komplexen Zahl Z

Die bisher verwendete Normalform $Z = a + b\mathrm{i}$ und die Polarform lassen sich nach den gleichen Gesetzen wie bei Vektoren ineinander umrechnen (vgl. Abschnitt 4.1)

$$r = \sqrt{a^2 + b^2} \tag{5.12}$$

$$\varphi = \arctan\left(\frac{a}{b}\right) \tag{5.13}$$

$$a = r \cdot \cos\varphi \tag{5.14}$$

$$b = r \cdot \sin\varphi \tag{5.15}$$

Beispiel

Auf die beiden komplexen Zahlen

$$Z = 5 + 7\mathrm{i}$$

und

$$Z = -3 - 5\mathrm{i}$$

sind die vier Grundrechenarten anzuwenden; des weiteren ist Z in die Polarform umzuwandeln.

$$(5+7i)+(-3-5i)=2+2i$$

$$(5+7i)-(-3-5i)=8+12i$$

$$(5+7i)*(-3-5i)=(-15+35)+(-25-21)i=20-46i$$

$$(5+7i)/(-3-5i)=\frac{-15-35}{9+25}+\frac{25-21}{9+25}i=-\frac{50}{34}+\frac{4}{34}i$$

$$Z_1=5+7i$$

$$r=\sqrt{25+49}=8{,}602$$

$$\varphi=\arctan\left(\frac{5}{7}\right)=35{,}54°$$

Die Polarform von Z_1 lautet also:

$$Z_1=8{,}602\,(\sin 35{,}54°+i\cos 35{,}54°)$$

Literaturhinweise

G: [19], S. 16 bis 20; G, W: [5], S. 400 bis 408; G, W: [13], S. 130 bis 146; G, W: [22], S. 151 ff; W: [19], S. 127 ff; W: [23]

Programmbeschreibung

Die Struktur des Programms *komplex* ist mit dem in Kapitel 4 beschriebenen Programm *Vektoralgebra* identisch. Neben den u. U. anderen Rechenvorschriften besteht ein wesentlicher Unterschied zwischen Programmen eigentlich nur darin, daß das Programm *komplex* nur zweidimensionale Rechnungen durchführen muß, während die Anzahl der Dimensionen des Vektorraumes im Programm *Vektoralgebra* frei wählbar ist. Das Programm hat demzufolge auch nur eine Länge von etwa 3,3 K, während Vektoralgebra etwa 5,1 K an Speicherplatz benötigt.

In den

Zeilen 50
bis 180 ist das Menü abgelegt, über das die verschiedenen Rechenarten angewählt werden können, die mit komplexen Zahlen möglich sind. Über die

Zeilen 300
bis 330 wird die erste komplexe Zahl eingegeben; die

Zeilen	400	
bis	440	werden bei Addition/Subtraktion und die
Zeilen	500	
bis	550	bei Multiplikation aufgerufen.
Zeile	600	
bis	660	ist schließlich zuständig für die Division von komplexen Zahlen. In den
Zeilen	700	
bis	1070	werden Polarform und Normalform ineinander umgerechnet; dieser Programmabschnitt ist praktisch identisch mit dem entsprechenden Abschnitt im Programm *Vektoralgebra*. Über die
Zeilen	1100	
bis	1160	kann zum Schluß noch die komplexe Zahl, die das Ergebnis der Rechnungen darstellt, ausgegeben werden.

```
10 REM Programm komplex
20 REM Rechenoperationen mit komplexen Zahlen
30 MODE 2
40 REM faellt aus wegen Bodennebel
50 CLS : REM Hauptmenue
60 PRINT"Was soll gemacht werden";:PRINT"__________
________________________________________________
________________"
70 PRINT:PRINT"- die 1.komplexe Zahll soll eingegebe
n werden";TAB(60);"(1)"
80 PRINT:PRINT"- eine komplexe Zahl soll addiert wer
den";TAB(60);"(2)"
90 PRINT:PRINT"- eine komplexe Zahl soll subtrahiert
 werden";TAB(60);"(3)"
100 PRINT:PRINT"- es soll mit einer komplexen Zahl m
ultipliziert werden";TAB(60);"(4)"
```

```
110 PRINT:PRINT"- es soll durch eine komplexen Zahl
dividiert werden";TAB(60);"(5)"
120 PRINT:PRINT"- Polarform und Normalform sollen um
gerechnet werden";TAB(60);"(6)"
130 PRINT:PRINT"- die berechnete Zahl soll ausgegebe
n werden";TAB(60);"(7)"
140 PRINT:PRINT"- Schluss";TAB(60);"(8)"
150 INPUT was
160 IF was <1 OR was>8 THEN 50
170 ON was GOSUB 300,400,400,500,600,700,1100,1200

180 GOTO 50
300 REM HP/UP Eingabe der 1.Zahl
310 GOSUB 1500 : REM Eingabe nch .2
320 x1=x2 : y1=y2
330 RETURN
400 REM HP/UP fuer + und -
410 GOSUB 1500 : REM Eingabe der 2. Zahl
420 IF was = 2 THEN x1=x1+x2 : y1=y1+y2
430 IF was = 3 THEN x1=x1-x2 : y1=y1-y2
440 RETURN
500 REM HP/UP Multiplikation
510 GOSUB 1500 : REM Multiplikatoreingabe
520 x3 = x1*x2 - y1*y2
530 y3 = x1*y2 + y1*x2
540 x1=x3 : y1=y3
550 RETURN
600 REM HP/UP Division
610 GOSUB 1500 : REM Eingabe des Divisors
615 y2=-y2
620 x3=x1*x2 - + y1*y2
```

```
630 y3 = x1*y2 + y1*x2
640 nenner = x2*x2 + y2*y2
650 x1=x3/nenner : y1=y3/nenner
660 RETURN
700 REM Umrechnungen von Polarform und Normalform
710 CLS
720 PRINT"Bei einer Umrechnung von der Normalform in
 die Polarform muessen die Faktoren   der Normalform
 dem Programm bereits in x1
und y1 vorliegen."
730 PRINT:PRINT"Bei der umgekehrten Rechnung wird di
e berechnete Normalform nach x1 und y1      gespeich
ert.":PRINT"________________
-------------------------------------------------------
____________"
740 PRINT:PRINT"Was soll umgerechnet werden ?"
750 PRINT:PRINT:PRINT"- Normalform in Polarform    (
1)"
760 PRINT:PRINT"- Polarform in Normalform    (2)"

770 INPUT was
780 IF was=1 THEN 900
800 REM Umrechnung Polarform in Normalform
810 CLS
820 DEG
830 INPUT"Betrag : ";betrag
840 PRINT:INPUT"Winkel : ";winkel
850 x1=betrag*COS(winkel)
860 y1=betrag*SIN(winkel)
870 GOSUB 1000 : REM Ausgabe
880 RETURN
```

```
900 REM Umrechnung Normalform in Polarform
910 DEG
940 betrag = SQR(x1*x1 + y1*y1)
950 winkel = ATN(y1/x1)
960 GOSUB 1000 : REM Ausgabe
970 RETURN
1000 REM Ausgabe fuer Umrechnungen Polar in Normalfo
rm   und umgekehrt
1010 CLS
1020 PRINT"Realteil : ";x1,"Imaginaerteil : ";y1
1030 PRINT:PRINT"Betrag : ";betrag,"Phi : ";winkel;"
grad"
1040 PRINT:PRINT"(Phi u.U. nicht richtig, da atan ni
cht eindeutig)"
1050 LOCATE 1,23 : PRINT"Weiter mit beliebiger Taste
"
1060 IF INKEY$="" THEN 1060
1070 RETURN
1100 REM HP/UP Ausgabe einer komplexen Zahl
1110 CLS
1120 PRINT"Realteil      : ";x1
1130 PRINT"Imaginaerteil : ";y1
1140 LOCATE 1,23 : PRINT"Weiter mit beliebiger Taste
"
1150 IF INKEY$="" THEN 1150
1160 RETURN
1200 REM HP/UP Schluss
1210 CLS
1220 LOCATE 30,10 : PRINT"Auf Wiedersehen"
1230 LOCATE 1,23
1240 END
```

```
1500 REM UP Eingabe einer komplexen Zahl
1510 CLS
1520 INPUT"Realteil       : ";x2
1530 INPUT"Imaginaerteil : ";y2
1540 RETURN
```

6
Determinanten

6.1 Eigenschaften von Determinanten

Grundlagen

Eine zweireihige Determinante ist dadurch definiert, daß dem quadratischen Zahlenschema

$$\begin{matrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{matrix}$$

der Zahlenwert

$$\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}a_{22} - a_{12}a_{21} \qquad (6.1.1)$$

zugeordnet wird. a_{ik} sind die Elemente der Determinante, wobei i die Zeile und k die Spalte angibt. Die Linie $a_{11}a_{22}$ nennt man die Hauptdiagonale, die Linie $a_{12}a_{21}$ Nebendiagonale.

Aus obiger Definition kann man die Richtigkeit der folgenden Regeln herleiten:

Regel 1

Der Wert einer Determinante bleibt unverändert, wenn man an der Hauptdiagonalen spiegelt.

$$\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = \begin{vmatrix} a_{11} & a_{21} \\ a_{12} & a_{22} \end{vmatrix} = a_{11}a_{22} - a_{12}a_{21}$$

Das Spiegeln an der Hauptdiagonalen ist auch bekannt als Stürzen; Zeilen und Spalten werden hierbei einfach vertauscht. Somit ist auch bewiesen, daß alle für Zeilen hergeleiteten Eigenschaften auch für Spalten gelten und umgekehrt.

Regel 2

Eine Determinante wird mit einem Faktor multipliziert, indem man die Elemente einer beliebigen Zeile oder Spalte mit ihm multipliziert.

$$k * (a_{11}a_{22} - a_{21}a_{12}) = ka_{11}a_{22} - ka_{21}a_{12}$$
$$= \begin{vmatrix} ka_{11} & a_{12} \\ ka_{21} & a_{22} \end{vmatrix}$$

Regel 3

Der Wert einer Determinante bleibt unverändert, wenn man zu einer Zeile ein beliebiges Vielfaches einer anderen Zeile addiert.

$$\begin{vmatrix} a_{11} + ta_{21} & a_{12} + ta_{22} \\ a_{21} & a_{22} \end{vmatrix} = (a_{11} + ta_{21})a_{22} - a_{21}(a_{12} + ta_{22})$$
$$= a_{11}a_{22} + ta_{21}a_{22} - a_{21}a_{12} - ta_{22}a_{21}$$
$$= a_{11}a_{22} - a_{21}a_{12}$$

Regel 4

Der Wert einer Determinante ist gleich 0, wenn eine Zeile ein Vielfaches einer anderen Zeile ist.

$$\begin{vmatrix} a_{11} & a_{12} \\ ka_{11} & ka_{12} \end{vmatrix} = a_{11}ka_{12} - ka_{11}a_{12} = 0$$

Wenn alle Elemente der Determinante gleich 0 sind, kann man ebenfalls obige Regel anwenden.

Regel 5

Eine Determinante, die in der Hauptdiagonalen nur Einsen, sonst nur Nullen enthält, hat den Wert 1.

$$\begin{vmatrix} 1 & 1 \\ 0 & 1 \end{vmatrix} = 1 * 1 - 0 * 0 = 1$$

Alle bisher hergeleiteten Regeln gelten nur für zweireihige Determinanten.

Genauso gibt es aber auch drei- und mehrreihige Determinanten:

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = \text{dreireihige Determinante}$$

Bei drei- bzw. mehrreihigen Determinanten hat man sogenannte Unterdeterminanten definiert, die jeweils einem Element zugeordnet sind; die zu einem Element a_{ik} gehörige Unterdeterminante erhält man durch Streichen der i-ten Zeile und k-ten Spalte. Diese Definition wird benutzt, um den Wert einer mehrreihigen Determinante zu bestimmen. Hierzu werden nacheinander alle Elemente einer beliebigen Zeile i oder Spalte k genommen, das Produkt dieser Elemente mit der jeweiligen Unterdeterminante gebildet, wobei das Vorzeichen bei + anfängt und dann bei jedem Glied wechselt, und schließlich die Summe gebildet.

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = a_{11} \begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} - a_{12} \begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix} + a_{13} \begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix}$$

Dieses etwas komplizierte Verfahren nennt man Entwickeln einer Determinante nach der Spalte k bzw. Zeile i; als Ergebnis erhält man Determinanten, die jeweils eine Reihe weniger besitzen als die Ursprungsdeterminante. Man muß also so oft entwickeln, bis nur noch zweireihige Determinanten vorhanden sind, von denen man nach (6.1.1) den Wert bestimmen kann. Ein anderes Verfahren der Eigenwertbestimmung ist in Abschnitt 6.2 beschrieben.

Regel 6

Alle für 2reihige Determinanten aufgestellten Sätze gelten auch für die drei- und mehrreihigen Determinanten. Für den Beweis dieses Satzes sei auf die Literatur [9] verwiesen.

Regel 7

Hat eine Determinante oberhalb (und/oder unterhalb) der Hauptdiagonalen nur Nullen, so ist ihr Wert gleich dem Produkt der diagonalen Elemente.

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22} & a_{23} \\ 0 & 0 & a_{33} \end{vmatrix} = a_{11} \begin{vmatrix} a_{22} & a_{23} \\ 0 & a_{33} \end{vmatrix} - 0 \begin{vmatrix} a_{12} & a_{13} \\ 0 & a_{33} \end{vmatrix} + 0 \begin{vmatrix} a_{12} & a_{13} \\ a_{22} & a_{23} \end{vmatrix}$$

$$= a_{11}(a_{22}a_{33} - 0 * a_{23}) = a_{11}a_{22}a_{33}$$

Die Entwicklung wird zweckmäßigerweise nach der 1. Spalte oder 3. Zeile vorgenommen, alle anderen Entwicklungen führen aber zum gleichen Ergebnis.

Beispiel

Der Wert folgender Determinante soll berechnet werden:

$$\begin{vmatrix} 2 & 4 & -2 \\ 6 & -1 & 5 \\ 3 & -2 & 6 \end{vmatrix} = 2 \begin{vmatrix} -1 & 5 \\ -2 & 6 \end{vmatrix} - 4 \begin{vmatrix} 6 & 5 \\ 3 & 6 \end{vmatrix} - 2 \begin{vmatrix} 6 & -1 \\ 3 & -2 \end{vmatrix}$$

$$= 2(-1*6 + 2*5) - 4(6*6 - 3*5) - 2(-2*6 + 1*3)$$

$$= 2*4 - 4*21 + 2*9$$

$$= -58$$

Literaturhinweise

G, W: [5], S. 116 ff; G, W: [9], S. 81 ff; G, W: [13], S. 281 ff; G, W: [16], S. 87 ff; G, W: [19], S. 37 ff; G, W: [22], S. 84 ff.

6.2 Eigenwertbestimmung

Grundlagen

Das Entwicklungsverfahren zur Bestimmung des Eigenwertes einer Determinante wird leicht unübersichtlich, wenn es sich um mehrreihige Determinanten handelt; hier wendet man besser den sogenannten

Gaußschen Algorithmus an. Die Bestimmung des Eigenwertes geht dann folgendermaßen vor sich:

$$\text{Determinante} = \begin{vmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{vmatrix}$$

Die erste Zeile wird mit $-a_{k1}/a_{11}$ multipliziert und zur k-ten Zeile addiert ($k = 2,3,\dots, n$) (1 fehlt!). Der Wert der Determinante bleibt hierbei nach Regel 3, Abschnitt 6.1, unverändert. Das Ergebnis ist folgende Determinante:

$$\begin{vmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ 0 & a'_{22} & \dots & a'_{2m} \\ \vdots & \vdots & & \vdots \\ 0 & a'_{n2} & \dots & a'_{nm} \end{vmatrix}$$

Anschließend nimmt man die zweite Zeile mit $-a'_{k2}/a'_{22}$ mal und addiert sie zur k-ten Gleichung ($k = 1,3,\dots, n$) (jetzt fehlt 2!). Dieses Verfahren wendet man n-mal an. Das Ergebnis ist die Determinante

$$\begin{vmatrix} a^*_{11} & 0 & \dots & 0 \\ 0 & a^*_{22} & \dots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & a^*_{nm} \end{vmatrix}$$

Der Wert dieser Determinante ist aber nach Regel 7, Abschnitt 6.1 gleich dem Produkt der Elemente der Hauptdiagonalen.

Beispiel

Zur Verdeutlichung soll die gleiche Determinante wie im Abschnitt 6.1 durchgerechnet werden.

$$\begin{vmatrix} 2 & 4 & -2 \\ 6 & -1 & 5 \\ 3 & -2 & 6 \end{vmatrix} \qquad n = 3$$

1. Schritt: $k = 2$: $-\frac{a_{k1}}{a_{11}} = -\frac{a_{21}}{a_{11}} = -\frac{6}{2} = -3$

$$\begin{vmatrix} 2 & 4 & -2 \\ 0 & -13 & 11 \\ 3 & -2 & 6 \end{vmatrix}$$

$k = 3$: $-\frac{a_{k1}}{a_{11}} = -\frac{a_{31}}{a_{11}} = -\frac{3}{2}$

$$\begin{vmatrix} 2 & 4 & -2 \\ 0 & -13 & 11 \\ 0 & -8 & 9 \end{vmatrix}$$

2. Schritt: $k = 1$: $-\frac{a'_{k2}}{a'_{22}} = -\frac{a'_{12}}{a'_{22}} = -\frac{4}{-13} = \frac{4}{13}$

$$\begin{vmatrix} 2 & 0 & \frac{18}{13} \\ 0 & -13 & 11 \\ 0 & -8 & 9 \end{vmatrix}$$

$k = 3$: $-\frac{a'_{k2}}{a'_{22}} = -\frac{a'_{32}}{a'_{23}} = -\frac{-8}{-13} = -\frac{8}{13}$

$$\begin{vmatrix} 2 & 0 & \frac{18}{13} \\ 0 & -13 & 11 \\ 0 & 0 & \frac{29}{13} \end{vmatrix}$$

An dieser Stelle könnte man bereits aufhören, da die Bedingungen für die Regel 7 erfüllt sind. Zur Verdeutlichung wird jedoch der dritte Schritt auch noch durchgeführt.

3. Schritt: $k = 1$: $-\frac{a'_{13}}{a'_{33}} = -\frac{\frac{18}{13}}{\frac{29}{13}} = -\frac{18}{29}$

$$\begin{vmatrix} 2 & 0 & 0 \\ 0 & -13 & 11 \\ 0 & 0 & \frac{29}{13} \end{vmatrix}$$

$$k = 2: \quad -\frac{a'_{23}}{a'_{33}} = -\frac{11}{\frac{29}{13}} = -\frac{143}{29}$$

$$\begin{vmatrix} 2 & 0 & 0 \\ 0 & -13 & 0 \\ 0 & 0 & \frac{29}{13} \end{vmatrix}$$

Der Eigenwert der Determinante ist hiernach also

$$2 * (-13) * \frac{29}{13} = -58$$

Wie man sieht, ist dieses Verfahren für dreireihige Determinanten sehr viel komplizierter als das Entwicklungsverfahren; da es aber einem ganz klaren Schema folgt, läßt es sich leicht auf einem Rechner anwenden.

Literaturhinweise

G, W: [5], S. 116 ff; G, W: [9], S. 81 ff; G, W: [13], S. 281 ff; G, W: [16], S. 87 ff; G, W: [19], S. 37 ff; G, W: [22], S. 84 ff.

Programmbeschreibung

Auch für das Programm *determinante* wurde die in den Kapiteln 4 und 5 beschriebene Programmstruktur beibehalten, obwohl es hier vielleicht aufgrund der geringen Anzahl von Rechenoperationen nicht notwendig gewesen wäre; es wird jedoch auf diese Weise eine bessere Übersichtlichkeit erreicht.

		Nach Eingabe der Determinantengröße in
Zeile	30	und der Dimensionierung des Hauptfeldes *dadd* und des Berechnungsfeldes *d* (vgl. Kapitel 4 und 5) in
Zeile	40	erfolgt in
Zeile	50	
bis	200	die Ausgabe des Menüs. Im Gegensatz zu den erwähnten Programmen erfolgt die Eingabe der Codezahlen hier nicht direkt; vielmehr werden Zeichen eingegeben, die in logischer Verbindung zu der durchzuführenden Rechnung stehen. Diese Zeichen werden in den

Zeilen 140	
bis 180	dann in Codezahlen umgewandelt, so daß
Zeile 190	das entsprechende Unterprogramm auswählen kann. Da sich die Unterprogramme für Eingabe und Ausgabe ähneln, werden sie im Unterprogramm in den
Zeilen 300	
bis 400	zusammengefaßt; wo Unterscheidungen notwendig sind, geschieht dies aufgrund der Codezahl. Entsprechend erfolgt in
Zeile 500	
bis 560	die Multiplikation mit einem Faktor. Obwohl hier prinzipiell eine beliebige Spalte oder Zeile der Determinante mit dem Faktor multipliziert werden kann, wählt das Programm immer die erste Spalte; soll dies anders sein, müssen die entsprechenden Indizes in
Zeile 540	geändert werden. Ab
Zeile 700	erfolgt die Eigenwertbestimmung der Determinante, die in *dadd* gespeichert ist. Um diese für weitere Rechnungen noch zur Verfügung zu haben, erfolgt zunächst in
Zeile 710	
bis 750	die Umspeicherung nach *d*. Die Berechnung der Lösungsdeterminanten, in der nur noch in der Hauptdiagonalen Werte stehen, erfolgt in den
Zeilen 1100	
bis 1200	Die Lösung erhält man durch Multiplikation der einzelnen Faktoren der Hauptdiagonalen. Dies wird in
Zeile 1220	
bis 1260	durchgeführt. Die
Zeilen 800	
bis 1040	stellen Unterprogramme dar, die von den anderen Teilprogrammen benutzt werden; ihre Bedeutung kann den Kommentarzeilen entnommen werden.

Programmlänge: etwa 2,7 K.

```
10 REM HP Determinantenrechnungen
20 MODE 2
30 PRINT"Wieviele Zeilen bzw. Spalten hat (haben) di
e Determinante(n) ";:INPUT danz
40 DIM d(danz,danz),dadd(danz,danz)
50 REM Hauptmenue
60 CLS
70 PRINT"Was soll gemacht werden ?" : PRINT "_______
____________________________________________________
_____________________"
80 PRINT:PRINT"- die 1.Determinante soll eingegeben
werden";TAB(70);"(1)"
90 PRINT:PRINT"- es soll mit einem Faktor multiplizi
ert werden";TAB(70);"(*)"
100 PRINT:PRINT"- die berechnete Determinante soll a
usgegeben werden";TAB(70);"(a)"
110 PRINT:PRINT"- der Eigenwert der berechneten Dete
rminante soll bestimmt werden";TAB(70);"(e)"
120 PRINT:PRINT"- es soll endlich Schluss gemacht we
rden";TAB(70);"(s)"
130 INPUT a$
140 IF a$="1" THEN was =1
150 IF a$="*" THEN was = 2
160 IF a$="a" THEN was = 3
170 IF a$="s" THEN was = 4
180 IF a$="e" THEN was = 5
190 ON was GOSUB 300,500,300,600,700
200 GOTO 50
300 REM HP/UP fuer 1 und a
310 IF was = 3 THEN 330
320 GOSUB 900 : REM Eingabe der Determinante
```

```
330 CLS
340 FOR k=1 TO danz
350 FOR l=1 TO danz
360 ON was GOSUB 810,1320,830
370 NEXT l
380 NEXT k
390 IF was <> 3 THEN 420
400 LOCATE 1,23 : PRINT"Weiter mit beliebiger Taste"
410 IF INKEY$="" THEN 410
420 RETURN
500 REM UP/HP Multiplikation mit einem Faktor
510 CLS
520 INPUT "Faktor : ";fak
530 FOR k=1 TO danz
540 dadd(k,1) = dadd(k,1)*fak
550 NEXT k
560 RETURN
600 REM UP/HP Ende
610 CLS
620 LOCATE 20,10 : PRINT"Ich hab auch keine Lust meh
r"
630 LOCATE 1,23
640 END
700 REM UP/HP Eigenwertbestimmung
710 FOR k=1 TO danz
720 FOR l=1 TO danz
730 d(k,l) = dadd(k,l)
740 NEXT l
750 NEXT k
760 GOSUB 1100 : REM Eigenwertbestimmung
770 CLS : PRINT"Eigenwert : ";dew
```

```
780 RETURN
800 REM UP's fuer l und a
810 dadd(k,l) = d(k,l) : REM Umspeicherung
820 RETURN
830 LOCATE l*10-9 , k : PRINT dadd(k,l)
840 RETURN
900 REM UP Determinanteneingabe
910 WINDOW #1,1,80,5,10
920 CLS
930 PRINT"Geben Sie die Determinante zeilenweise ein
"
940 FOR k=1 TO danz
950 LOCATE 1,3 : PRINT k;".te Zeile":PRINT
960 CLS#1
970 FOR l=1 TO danz
980 PRINT#1,"d(";k;",";l;") : ";:INPUT#1, d(k,l)
990 NEXT l
1000 NEXT k
1010 LOCATE 1,20 : INPUT"Eingabe i.o ( /n) ";a$
1020 IF a$="" THEN 1040
1030 IF a$="n" THEN 900 ELSE 1010
1040 RETURN
1100 REM UP Eigenwertbestimmung
1110 REM verwendet wird der Gauss'sche Algorithmus
1120 FOR m=1 TO danz
1130 FOR k=1 TO danz
1140 IF k=m THEN 1190
1150 al=-d(k,m)/d(m,m)
1160 FOR l=1 TO danz
1170 d(k,l)=d(k,l) + d(m,l)*al
1180 NEXT l
```

```
1190 NEXT k
1200 NEXT m
1210 REM Loesungsmatrix ist fertig
1220 REM dew wird durch Multiplikation der Hauptdiag
onalen bestimmt
1230 LET dew=1
1240 FOR i=1 TO danz
1250 LET dew=dew*d(i,i)
1260 NEXT i
1270 CLS
1280 PRINT"Eigenwert : ";dew
1290 LOCATE 1,23 : PRINT"Weiter mit beliebiger Taste
"
1300 IF INKEY$="" THEN 1300
1310 RETURN
1320 REM
```

7
Matrizen

Grundlagen

Mit den Determinanten verwandt sind die Matrizen; auch hier handelt es sich um Zahlenschemata, die jedoch nicht mehr zwangsläufig quadratisch sein müssen. Das Produkt aus der Zeilenanzahl i und der Spaltenanzahl k nennt man auch Grad der Matrix.

$$grad = (i * k)$$

Gleich sind zwei Matrizen nur dann, wenn sie in allen ihren Koeffizienten übereinstimmen. Im Gegensatz zu Determinanten lassen sich Matrizen z.B. durch Addition miteinander verknüpfen:

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ \vdots & & & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix} + \begin{pmatrix} b_{11} & b_{12} & \dots & b_{1n} \\ \vdots & \vdots & & \vdots \\ b_{m1} & b_{m2} & \dots & b_{mn} \end{pmatrix}$$

$$= \begin{pmatrix} a_{11} + b_{11} & a_{12} + b_{12} & \dots & a_{1n} + b_{1n} \\ \vdots & \vdots & & \vdots \\ a_{m1} + b_{m1} & a_{m2} + b_{m2} & \dots & a_{mn} + b_{mn} \end{pmatrix} \tag{7.1}$$

Matrizen werden also koeffizientenweise addiert; das gleiche gilt für die Subtraktion. Daraus folgt auch, daß Matrizen nur dann addiert bzw. subtrahiert werden können, wenn sie in Zeilen- und Spaltenzahl übereinstimmen. Ähnliches gilt für die Multiplikation einer Matrix mit einem Faktor:

$$k * \begin{pmatrix} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \dots & a_{mn} \end{pmatrix} = \begin{pmatrix} ka_{11} & \dots & ka_{1n} \\ \vdots & & \vdots \\ ka_{m1} & \dots & ka_{mn} \end{pmatrix} \tag{7.2}$$

Auch die Faktormultiplikation wird demnach koeffizientenweise durchgeführt.

Komplizierter wird es, wenn zwei Matrizen multipliziert werden sollen. Dies ist nur möglich, wenn die Spaltenanzahl k der ersten Matrix mit der Zeilenanzahl i der zweiten Matrix übereinstimmt. Wenn r die Zeilenanzahl der ersten Matrix und s die Spaltenanzahl der zweiten Matrix bezeichnet, so ist das Ergebnis eine $(r * s)$-Matrix.

$$\begin{pmatrix} a_{11} & \dots & a_{1k} \\ \vdots & & \vdots \\ a_{r1} & \dots & a_{rk} \end{pmatrix} * \begin{pmatrix} b_{11} & \dots & b_{15} \\ \vdots & & \vdots \\ b_{i1} & \dots & b_{is} \end{pmatrix} = \begin{pmatrix} c_{11} & \dots & c_{1s} \\ \vdots & & \vdots \\ c_{r1} & \dots & c_{rs} \end{pmatrix}$$

Das Glied c_{11} der Ergebnismatrix erhält man, indem man jeweils das Produkt bildet aus den Koeffizienten der ersten Zeile von Matrix a mit den entsprechenden Koeffizienten der ersten Spalte von Matrix b; c_{11} ist die Summe aller Produkte.

Für c_{12} multipliziert man die Elemente der ersten Zeile von Matrix a mit den Elementen der zweiten Spalte von Matrix b und bildet die Summe. Entsprechend verfährt man mit den übrigen Gliedern der Ergebnismatrix; für c_{34} z.B. nimmt man die Elemente der dritten Zeile von Matrix a und die der vierten Spalte von Matrix b.
Zusammengefaßt bedeutet dies:

$$c_{mn} = \sum_{t=1}^{k} a_{mt} b_{tn} \tag{7.3}$$

$m = 1, \dots, r$
$n = 1, \dots, s$
$i = k!$
r = Zeilenanzahl Matrix a
k = Spaltenanzahl Matrix a
i = Zeilenanzahl Matrix b
s = Spaltenanzahl Matrix b

Für die Herleitung von Regel (7.3) sei auf die Literatur [22] verwiesen.

Genau wie eine Determinante läßt sich eine Matrix an der Hauptdiagonalen spiegeln; dies nennt man hier auch transponieren. Aus einer $(i * k)$-Matrix entsteht eine $(k * i)$-Matrix; Zeilen und Spalten werden vertauscht. Per Definition heißt eine quadratische Matrix, deren Determinante den Wert 0 hat (vgl. Abschnitt 6.1), singulär, anderenfalls heißt sie regulär.

Für reguläre Matrizen gibt es neben den beschriebenen Verknüpfungen noch eine weitere Umformung, die des Öfteren in der Technik angewendet werden muß, das sogenannte Invertieren.
Für die zu a inverse Matrix a^{-1} gilt:

$$a * a^{-1} = a^{-1} * a = \mathrm{E} \tag{7.4}$$

E ist die Einheitsmatrix, sie enthält in der Hauptdiagonalen nur Einsen, alle anderen Glieder sind 0. Zur Aufstellung der inversen Matrix schreibt man zunächst eine gleichrangige Einheitsmatrix E neben die umzuformende Matrix a. Man betrachte dann beide Matrizen als Determinanten und wendet die elementaren Umformungen nach Kapitel 6 an, bis die Determinante der Matrix a eine Einheitsdeterminante darstellt. Zweckmäßigerweise verwendet man hierzu den Gaußschen Algorithmus wie in Abschnitt 6.2, bis nur noch in der Hauptdiagonalen Werte stehen. Diese zieht man dann durch Division vor die Determinante, so daß die Einheitsdeterminante entsteht. Alle Rechenoperationen, die man an a ausführt, führt man parallel dazu auch an E aus. Wenn a zu E geworden ist, steht an der Stelle, an der die Einheitsmatrix stand, die Inverse a^{-1}. Prüfen kann man dies, indem man a und a^{-1} miteinander multipliziert; das Ergebnis muß nach Gleichung (7.4) die Einheitsmatrix E sein.

Beispiele

$$a = \begin{pmatrix} 3 & 5 & 7 \\ 4 & 2 & 9 \\ 4 & 3 & 8 \end{pmatrix} \qquad b = \begin{pmatrix} 1 & 2 & 9 \\ 8 & 0 & 4 \\ 3 & 7 & 3 \end{pmatrix}$$

$$a + b = \begin{pmatrix} 4 & 7 & 16 \\ 12 & 2 & 13 \\ 7 & 10 & 11 \end{pmatrix} \qquad a - b = \begin{pmatrix} 2 & 3 & -2 \\ -4 & 2 & 5 \\ 1 & -4 & 5 \end{pmatrix}$$

$$3 * a = \begin{pmatrix} 9 & 15 & 21 \\ 12 & 6 & 27 \\ 12 & 9 & 24 \end{pmatrix}$$

Matrizenmultiplikation $a * b$

$r = k = i = s = 3$

$$c_{11} = \sum a_{1t} b_{t1} = 3*1 + 5*8 + 7*3 = 64$$
$$c_{12} = \sum a_{1t} b_{t2} = 3*2 + 5*0 + 7*7 = 55$$
$$c_{13} = \sum a_{1t} b_{t3} = 3*9 + 5*4 + 7*3 = 68$$
$$c_{21} = \sum a_{2t} b_{t1} = 4*1 + 2*8 + 9*3 = 47$$
$$c_{22} = \sum a_{2t} b_{t2} = 4*2 + 2*0 + 9*7 = 71$$
$$c_{23} = \sum a_{2t} b_{t3} = 4*9 + 2*4 + 9*3 = 71$$
$$c_{31} = \sum a_{3t} b_{t1} = 4*1 + 3*8 + 8*3 = 52$$
$$c_{32} = \sum a_{3t} b_{t2} = 4*2 + 3*0 + 8*7 = 64$$
$$c_{33} = \sum a_{3t} b_{t3} = 4*9 + 3*4 + 8*3 = 72$$

$\left(\sum \text{ bedeutet hier } \sum_{t=1}^{3}\right)$

Die Ergebnismatrix lautet also:

$$a * b = \begin{pmatrix} 64 & 55 & 68 \\ 47 & 71 & 71 \\ 52 & 64 & 72 \end{pmatrix}$$

Invertieren der Matrix a

Da der Gaußsche Algorithmus bereits in Abschnitt 6.2 ausführlich behandelt wurde, sind hier nur noch die Ergebnisse der einzelnen Schritte angegeben.

$$a = \begin{pmatrix} 3 & 5 & 7 \\ 4 & 2 & 9 \\ 4 & 3 & 8 \end{pmatrix} \qquad \mathrm{E} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 3 & 5 & 7 \\ 0 & -4{,}667 & -0{,}333 \\ 0 & -3{,}667 & -1{,}333 \end{pmatrix} \qquad \begin{pmatrix} 1 & 0 & 0 \\ -1{,}333 & 1 & 0 \\ -1{,}333 & 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 3 & 0 & 6,643 \\ 0 & -4,667 & -0,333 \\ 0 & 0 & -1,071 \end{pmatrix} \qquad \begin{pmatrix} 0,429 & 1,071 & 0 \\ -1,333 & 1 & 0 \\ -0,286 & -0,786 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 3 & 0 & 0 \\ 0 & -4,667 & 0 \\ 0 & 0 & -1,071 \end{pmatrix} \qquad \begin{pmatrix} -2,2 & -3,8 & 6,2 \\ -1,244 & 1,244 & -0,311 \\ -0,286 & -0,786 & 1 \end{pmatrix}$$

$$E = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad a^{-1} = \begin{pmatrix} -0,733 & -1,267 & 2,067 \\ 0,267 & -0,267 & 0,067 \\ 0,267 & 0,733 & -0,933 \end{pmatrix}$$

Multipliziert man a mit a^{-1}, so erhält man

$$a * a^{-1} = \begin{pmatrix} 1,001 & 0,002 & 0,002 \\ 0,001 & 1,002 & 0,002 \\ 0,001 & 0,002 & 1,002 \end{pmatrix}$$

Die Abweichungen von der Einheitsmatrix sind auf Rundungsfehler zurückzuführen, da nur mit drei Dezimalstellen gerechnet wurde.

Literaturhinweise

G, W: [5], S. 537 bis 543; G, W: [9], S. 55 ff; G, W: [22], S. 236 ff; G, W: [25]; W: [1],; W: [16], S. 57 ff.

Programmbeschreibung

Das Programm Matrizenprogramme ist mit etwa 8K das längste in diesem Buch abgedruckte; Ursache sind die vielen und zum Teil komplizierten Rechenmöglichkeiten, die mit Matrizen möglich sind. Das Programm kann mit Matrizen rechnen, die aus bis zu zehn Zeilen bzw. Spalten bestehen; soll diese Zahl erhöht werden, müssen die dementsprechenden Dimensionierungen geändert werden.

Zeile 50
bis 200 enthält wiederum das Hauptmenü mit Eingabe der Codezahl und entsprechender Auswahl des Unterprogramms.

Zeile 300
bis 400 enthält unter Benutzung der

Zeilen 2100
bis 2280 die Eingabe der ersten Matrix; die Gründe für diese Vorgehensweise wurden bereits in Kapitel 4 erläutert.

Zeile 500
bis 650 ist zuständig für Addition und Subtraktion, wobei wie übrigens bei allen anderen Eingaben von Matrizen vor Eingabe der zu addierenden bzw. subtrahierenden Matrix das Eingabefeld gelöscht und neu dimensioniert wird. Da nur Matrizen mit gleicher Spalten- und Zeilenzahl addiert und subtrahiert werden können, erfolgt die entsprechende Überprüfung in

Zeile 530

Durch die

Zeilen 700
bis 820 kann die Matrix *mata*, in der alle Ergebnisse gespeichert werden, ausgegeben werden. Mit dem Unterprogramm in

Zeile 900
bis 980 wird die in *mata* befindliche Matrix mit einem Faktor multipliziert, wobei jedes Element mit diesem Faktor malgenommen wird. In den

Zeilen 1000
bis 1250 ist dagegen die Multiplikation mit einer Matrix beschrieben. Entsprechend der Rechenvorschrift wird in

Zeile 1040 geprüft, ob die Spaltenzahl von Matrix *a* mit der Zeilenzahl von Matrix *b* übereinstimmt; ist das nicht der Fall, erfolgt die entsprechende Ausgabe in

Zeile 1220 und der Rücksprung zum Menü.

Für die eigentliche Rechnung wird zunächst in den

Zeilen 1060
bis 1220 die Ergebnismatrix *matc* berechnet; diese wird in den

Zeilen 1130
bis 1190 nach *mata* gespeichert, damit sie mit dem vorher genannten Unterprogramm ausgegeben werden kann. Die

Zeilen 1300
bis 1420 transponieren *mata*, wobei die neue Matrix zunächst in *matb* gebildet wird und dann nach *mata* zurückgespeichert wird. In den

Zeilen 1500
bis 1860 wird die Inversion der Matrix *a* durchgeführt. Hierzu wird in

Zeilen 1520 zunächst geprüft, ob es sich um eine quadratische Matrix handelt. Ist das der Fall, wird *mata* in

Zeile 2370
bis 2430 zunächst in das Feld *d* gespeichert. Dies wird durchgeführt, weil anschließend zwischen

Zeile 2500
bis 2670 die Eigenwertbestimmung von *d* durchgeführt wird. Hierzu wird das Programm aus Kapitel 6 benutzt, so daß die Variablennamen angepaßt werden müssen.
Nur wenn der Eigenwert der Determinante von Matrix *a* ungleich 0 ist, wird in

Zeile 1550
bis 1740 die inverse Matrix in *matb* berechnet und in

Zeile 1750
bis 1790 nach *mata* zurückgespeichert. Ist der Eigenwert dagegen gleich 0, erfolgt in

Zeile 1810 die Ausgabe, daß es sich um eine singuläre Matrix handelt und dementsprechend der Rücksprung ins Menü.
Die Eigenwertbestimmung der Determinante einer Matrix kann über Codezahl 9 und dementsprechend über die

Zeilen 2000
bis 2050 und

Zeile 2300
bis 2670 auch alleine angewählt werden.
Bei diesem Programm handelt es sich, wie schon gesagt, um das gleiche Programm wie in Kapitel 6; für die Erläuterung des Ablaufes sei auf die Programmdokumentation dort verwiesen.

```
10 REM HP Matrizenprogramme
20 MODE 2
30 DIM mata(10,10),matb(10,10)
50 CLS : REM Hauptmenue
60 PRINT"Was soll gemacht werden ? " : PRINT"_______
--------------------------------------------------------
--------------------"
70 PRINT:PRINT"- Eingabe der 1.Matrix";TAB(50);"(1)"

80 PRINT:PRINT"- Addition einer Matrix";TAB(50);"(2)
"
90 PRINT:PRINT"- Subtraktion einer Matrix";TAB(50);"
(3)"
100 PRINT:PRINT"- Multiplikation mit einem Faktor";T
AB(50);"(4)"
110 PRINT:PRINT"- Multiplikation mit einer Matrix";T
AB(50);"(5)"
120 PRINT:PRINT"- Transponieren einer Matrix";TAB(50
);"(6)"
130 PRINT:PRINT"- Invertieren einer Matrix";TAB(50);
"(7)"
140 PRINT:PRINT"- Ausgabe der berechneten Matrix";TA
B(50);"(8)"
150 PRINT:PRINT"- Eigenwertbestimmung einer Matrix";
TAB(50);"(9)"
160 PRINT:PRINT"- Schluss ";TAB(50);"(10)"
170 INPUT was
180 IF was<1 OR was>10 THEN 50
190 ON was GOSUB 300,500,500,900,1000,1300,1500,700,
2000,1900
200 GOTO 50
```

```
300 REM HP/UP Eingabe der 1.Matrix
310 ERASE matb : DIM matb(10,10)
320 GOSUB 2100 : REM Eingabe Zeilen- und Spaltenzahl
330 GOSUB 2150 : REM Eingabe der Matrix
340 FOR k=1 TO bzeile
350 FOR i=1 TO bspalte
360 mata(k,i)=matb(k,i)
370 NEXT i
380 NEXT k
390 azeile = bzeile : aspalte = bspalte
400 RETURN
500 REM HP/UP + und -
510 ERASE matb : DIM matb(10,10)
520 GOSUB 2100
530 IF bzeile<>azeile OR bspalte<>aspalte THEN 620
540 GOSUB 2150 : REM Matrixeingabe
550 FOR k=1 TO bzeile
560 FOR i=1 TO bspalte
570 IF was=2 THEN mata(k,i)=mata(k,i) + matb(k,i)
580 IF was=3 THEN mata(k,i)=mata(k,i) - matb(k,i)
590 NEXT i
600 NEXT k
610 GOTO 650
620 CLS : PRINT"Es koennen nur Matrixen mit gleicher
 Spalten- und Zeilenzahl addiert bzw.       subtrahi
ert werden."
630 PRINT:PRINT"Weiter mit beliebiger Taste"
640 IF INKEY$="" THEN 640
650 RETURN
700 REM HP/UP Ausgabe
710 CLS
```

```
720 a2=0
730 FOR k=1 TO aspalte
740 FOR i=1 TO azeile
750 LOCATE (k-a2*6)*12-11,i : PRINT mata(i,k)
760 NEXT i
770 IF k=6 THEN GOSUB 800
780 IF k=6 THEN a2=a2+1 : CLS
790 NEXT k
800 LOCATE 1,23 : PRINT"Weiter mit beliebiger Taste"
810 IF INKEY$="" THEN 810
820 RETURN
900 REM HP/UP Faktormultipplikation
910 CLS
920 INPUT"Faktor : ";fak : CLS
930 FOR k=1 TO azeile
940 FOR i=1 TO aspalte
950 mata(k,i)=mata(k,i)*fak
960 NEXT i
970 NEXT k
980 RETURN
1000 REM HP/UP Multiplikation mit einer Matrix
1010 ERASE matb : DIM matb(10,10)
1020 GOSUB 2100 : REM eingabe neue Matrix
1030 GOSUB 2150 : REM dito
1040 IF aspalte<>bzeile THEN 1220
1050 DIM matc(azeile,bspalte)
1060 FOR k=1 TO azeile
1070 FOR i=1 TO bspalte
1080 FOR n=1 TO aspalte
1090 matc(k,i) = mata(k,n) * matb(n,i) + matc(k,i)
1100 NEXT n
```

```
1110 NEXT i
1120 NEXT k
1130 ERASE mata : DIM mata(10,10) : REM Nullsetzen v
on mata
1140 FOR k=1 TO azeile
1150 FOR i=1 TO bspalte
1160 mata(k,i) = matc(k,i)
1170 NEXT i
1180 NEXT k
1190 aspalte=bspalte
1200 ERASE matc
1210 GOTO 1250
1220 CLS : PRINT"Matrizenmultiplikationen sind nur m
oeglich, wenn die Spaltenanzahl von Matrix A = der Z
eilenanzahl von Matrix B ist
"
1230 LOCATE 1,23 : PRINT"Weiter mit beliebiger Taste
"
1240 IF INKEY$="" THEN 1240
1250 RETURN
1300 REM HP/UP Transponieren
1310 ERASE matb : DIM matb(10,10)
1320 FOR k=1 TO azeile
1330 FOR i=1 TO aspalte
1340 matb(i,k) = mata(i,k) : REM zunaechst Speicheru
ng nach mat b
1350 NEXT i
1360 NEXT k
1370 FOR k=1 TO azeile
1380 FOR i=1 TO aspalte
1390 mata(i,k) = matb(k,i)
```

```
1400 NEXT i
1410 NEXT k
1420 RETURN
1500 REM HP/UP Inversion einer Matrix
1510 ERASE matb : DIM matb(10,10)
1520 IF aspalte <> azeile THEN 1840
1530 GOSUB 2300 : REM Eigenwertbestimmung mit Pruefu
ng auf quadratische Matrix und Umspeicherung
1540 IF dew = 0 THEN 1810
1550 FOR k=1 TO aspalte : REM Aufbau einer Einheitsm
atrix in matb
1560 matb(k,k)=1
1570 NEXT k
1580 FOR m=1 TO aspalte
1590 FOR k=1 TO aspalte
1600 IF k=m THEN 1660
1610 a1= -mata(k,m)/mata(m,m)
1620 FOR i=1 TO aspalte
1630 mata(k,i) = mata(k,i) + mata(m,i)*a1
1640 matb(k,i) = matb(k,i) + matb(m,i)*a1
1650 NEXT i
1660 NEXT k
1670 NEXT m
1680 FOR k=1 TO aspalte
1690 a1=mata(k,k)
1700 FOR i=1 TO aspalte
1710 mata(k,i)=mata(k,i)/a1
1720 matb(k,i)=matb(k,i)/a1
1730 NEXT i
1740 NEXT k
1750 FOR k=1 TO aspalte
```

```
1760 FOR l=1 TO aspalte
1770 mata(k,l)=matb(k,l) : REM Umspeicherung der inv
ersen Matrix
1780 NEXT l
1790 NEXT k
1800 GOTO 1860
1810 CLS : PRINT"Die Matrix ist singulaer. Eine Inve
rtierung ist sinnlos.": LOCATE 1,23 : PRINT"Weiter m
it beliebiger Taste"
1820 IF INKEY$="" THEN 1820
1830 GOTO 1860
1840 CLS : PRINT"Invertierungen koennen nur von quad
ratischen Matrizen durchgefuehrt werden": LOCATE 1,2
3 : PRINT"Weiter mit beliebi
ger Taste"
1850 IF INKEY$="" THEN 1850
1860 RETURN
1900 REM UP/HP ende
1910 END
2000 REM HP/UP Eigenwertbestimmung mit Pruefung auf
quadratische Matrix und Umspeicherung
2010 GOSUB 2300 : REM ab da passiert's
2020 CLS : PRINT"Eigenwert : ";dew
2030 LOCATE 1,23 : PRINT"Weiter mit beliebiger Taste
"
2040 IF INKEY$="" THEN 2040
2050 RETURN
2100 REM UP Eingabe von bmat
2110 CLS
2120 INPUT"Wieviele Zeilen hat die Matrix";bzeile
2130 PRINT:INPUT"Wieviele Spalten hat die Matrix ";b
```

```
spalte
2140 RETURN
2150 REM Eingabe der Matrix
2160 WINDOW #1,1,80,8,18
2170 FOR k=1 TO bzeile
2180 CLS #1
2190 LOCATE 1,6 : PRINT k;".te Zeile"
2200 FOR i=1 TO bspalte
2210 PRINT#1,"mat(";k;",";i;") : ";:INPUT#1,matb(k,i
)
2220 NEXT i
2230 NEXT k
2240 LOCATE 1,23 : PRINT"Alle Eingaben i.o. ( /n) ";
:INPUT a$
2250 IF a$="" THEN 2280
2260 IF a$="n" THEN GOSUB 2100 ELSE 2240
2270 GOSUB 2150
2280 RETURN
2300 REM Pruefung auf quadratische Matrix fuer Eigen
wertbestimmung in 2500
2310 IF aspalte = azeile THEN 2370
2320 CLS
2330 PRINT"Eigenwertbestimmungen sind nur von Determ
inanten quadratischer Matrizen moeglich"
2340 LOCATE 1,23 : PRINT"Weiter mit beliebiger Taste
"
2350 IF INKEY$="" THEN 2350
2360 GOTO 50
2370 REM Umspeicherungen fuer die Eigenwertbestimmun
gen ab 2500
2380 DIM d(aspalte,aspalte)
```

```
2390 FOR k=1 TO aspalte
2400 FOR i=1 TO aspalte
2410 d(k,i)=mata(k,i)
2420 NEXT i
2430 NEXT k
2440 danz=aspalte
2450 GOSUB 2500 : REM Eigenwertbestimmung
2460 ERASE d
2470 RETURN
2500 REM Eigenwertbestimmung von Determinanten quadr
atischer Matrizen
2510 REM Benutzt wird der Gauss'sche Algorithmus
2520 FOR m=1 TO danz
2530 FOR k=1 TO danz
2540 IF k=m THEN 2590
2550 a1= -d(k,m)/d(m,m)
2560 FOR i=1 TO danz
2570 d(k,i)=d(k,i) + d(m,i)*a1
2580 NEXT i
2590 NEXT k
2600 NEXT m
2610 REM Loesungsdeterminante ist fertig
2620 REM Eigenwertbestimmung durch Multiplikation de
r Hauptdiagonalen
2630 dew=1
2640 FOR i=1 TO danz
2650 dew = dew * d(i,i)
2660 NEXT i
2670 RETURN
10000 CLS
10010 FOR i=1 TO azeile
```

```
10020 FOR k=1 TO azeile
10030 LOCATE k*10-9,i : PRINT matb(i,k)
10040 NEXT k
10050 NEXT i
```

8
Lineare Gleichungssysteme

8.1 Lösung mit dem Gaußschen Algorithmus

Grundlagen

Eine Anzahl von n Gleichungen, in denen insgesamt m Unbekannte vorkommen, heißen dann linear, wenn alle Unbekannten in der ersten Potenz auftreten. Lösen läßt sich dieses System, wenn

- ☐ die Anzahl der Gleichungen gleich der Anzahl der Unbekannten ist ($m = n$)
- ☐ die Gleichungen linear unabhängig sind (zwei Gleichungen sind dann linear abhängig, wenn die eine durch Multiplikation mit einem Faktor in die andere überführt werden kann)
- ☐ sich die Gleichungen nicht widersprechen

Ein solches Gleichungssystem hat die Form

$$\begin{matrix} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1m}x_m = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2m}x_m = b_2 \\ \vdots \qquad \vdots \qquad\qquad \vdots \qquad \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nm}x_m = b_n \end{matrix}$$

$x_1 \ldots x_m$ = Unbekannte

Indizes von a: 1. Index = $1 \ldots n$ = Zahl der Unbekannten
2. Index = $1 \ldots m$ = Zahl der Gleichungen

Wenn die Glieder $b_1, b_2, \ldots, b_n$ alle gleich 0 sind, heißt ein solches System homogen. In diesem Fall läßt es sich mit der vorliegenden Methode nicht lösen.

Die Lösung des Gleichungssystems erfolgt hier mit Hilfe des

Gaußschen Algorithmus. Dabei wird vorausgesetzt, daß das System lösbar ist, eine Prüfung auf Lösbarkeit erfolgt nicht.

Der prinzipielle Lösungsablauf ist wie in Abschnitt 6.2 bei der Bestimmung des Eigenwertes einer Determinante; da jedoch in der Ausführung einige Unterschiede bestehen, wird der Ablauf hier erneut in allen Einzelheiten erläutert.

Die Lösung geht folgendermaßen vor sich:
Die 1. Gleichung wird mit $-a_{k1}/a_{11}$ multipliziert und zur k-ten Gleichung addiert, wobei k von 2 bis n läuft (man beachte: 1 fehlt!).
Das Ergebnis sieht dann folgendermaßen aus:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1m}x_m &= b_1 \\ a'_{22}x_2 + \ldots + a'_{2m}x_m &= b'_2 \\ \vdots \qquad\qquad \vdots \quad &\quad \vdots \\ a'_{n2}x_2 + \ldots + a'_{nm}x_m &= b'_n \end{aligned}$$

Wie man sieht, existiert nur noch in der 1. Gleichung ein Glied mit der Unbekannten x_1.

Im zweiten Schritt läßt man alle Glieder mit x_2 bis auf $a'_{22}x_2$ verschwinden. Hierzu nimmt man die 2. Gleichung mit $-a'_{k2}/a'_{22}$ mal und addiert sie zur k-ten Gleichung ($k = 1, 3, \ldots, n$). Man beachte, daß diesmal bei k die 2 fehlt, es handelt sich um den 2. Schritt.

Mit dem neuen Gleichungssystem verfährt man dann wieder entsprechend, bis schließlich n Gleichungen vorliegen, die jeweils nur noch eine Unbekannte enthalten. Diese lassen sich dann leicht lösen.

Beispiel

Gleichung I: $2x_1 + x_2 - x_3 = -2$
Gleichung II: $x_1 - 3x_2 - 2x_3 = -5$
Gleichung III: $-x_1 - 4x_2 + x_3 = 7$

1. Schritt

$$k = 2: \quad -\frac{a_{n1}}{a_{11}} = -\frac{a_{21}}{a_{11}} = -\frac{1}{2}$$

Gleichung I multipliziert mit $-1/2$ und zu Gleichung II addiert, ergibt als neue Gleichung II′:

Gleichung II': $-3{,}5x_2 - 1{,}5x_3 = -4$

$$k = 3: \quad -\frac{a_{k1}}{a_{11}} = -\frac{a_{31}}{a_{11}} = -\frac{-1}{2} = 1/2$$

Jetzt wiederum Gleichung I multipliziert mit 1/2 und addiert zu Gleichung III ergibt als Gleichung III':

Gleichung III': $-3{,}5x + 0{,}5x = 6$

Als neues Gleichungssystem ergibt sich somit:

Gleichung I: $2x_1 + x_2 - x_3 = -2$
Gleichung II': $-3{,}5x_2 - 1{,}5x_3 = -4$
Gleichung III': $-3{,}5x_2 + 0{,}5x_3 = 6$

2. Schritt:

$$k = 1: \quad -\frac{a'_{k2}}{a'_{22}} = -\frac{a'_{12}}{a'_{22}} = -\frac{1}{-3{,}5} = \frac{1}{3{,}5}$$

Gleichung II' multipliziert mit 1/3,5 und addiert zu Gleichung I ergibt:

Gleichung I': $2x_1 - \frac{5}{3{,}5}x_3 = -\frac{11}{3{,}5}$

$$k = 3: \quad -\frac{a'_{k2}}{a'_{22}} = -\frac{a'_{32}}{a'_{22}} = -\frac{-3{,}5}{-3{,}5} = -1$$

Gleichung III'': $2x_3 = 10$

Damit erhält man nach dem zweiten Schritt als neues Gleichungssystem:

Gleichung I': $2x_1 \qquad - \frac{5}{3{,}5}x_3 = -\frac{11}{3{,}5}$
Gleichung II': $-3{,}5x_2 - 1{,}5x_3 = -4$
Gleichung III'': $2x_3 = 10$

An dieser Stelle würde man normalerweise mit dem Einsetzverfahren weitermachen; da das Programm jedoch den Gaußschen Algorithmus bis zum Ende durchführt, sei dies auch hier zur Verdeutlichung gemacht.

3. Schritt

$$k = 1: \quad -\frac{a''_{k3}}{a''_{33}} = -\frac{a'_{13}}{a''_{33}} = -\frac{-\frac{5}{3,5}}{2} = 5/7$$

Gleichung I″: $2x_1 = 4$

$$k = 2: \quad -\frac{a''_{k-3}}{a''_{33}} = -\frac{a''_{23}}{a''_{33}} = -\frac{-1,5}{2} = 3/4$$

Gleichung II″: $-3,5x_2 = 3,5$

Das zusammengefaßte, endgültige Gleichungssystem lautet also:

Gleichung I″: $2x_1 = 4$
Gleichung II″: $-3,5x_2 = 3,5$
Gleichung III″: $2x_3 = 10$

Daraus folgen die einzelnen Unbekannten als:

$$x_1 = 2$$
$$x_2 = -1$$
$$x_3 = 5$$

Setzt man diese Werte zur Probe in Gleichung I ein, so erhält man:

$$2x_1 + x_2 - x_3 = -2$$
$$2*2 + (-1) - 5 = -2$$
$$-2 = -2 \qquad \text{w.d.R.d.R.b.w.}$$

(womit die Richtigkeit der Rechnung bewiesen wäre)

Literaturhinweise

G, W: [5], S. 116 ff; G, W: [9], S. 81 ff; G, W: [13], S. 281 ff; G, W: [16], S. 87 ff; G, W: [19], S. 37 ff; G, W: [22], S. 84 ff.

Programmbeschreibung

Bei der Eingabe des Gleichungssystems ist es wichtig, daß die Gleichungen in der Form vorliegen, wie sie unter Grundlagen angegeben ist; das Restglied, d.h. der Faktor ohne Unbekannte, steht rechts vom Gleichheitszeichen. Falls eine Unbekannte in einer Gleichung nicht vorkommt, muß an der entsprechenden Stelle der Faktor 0 angegeben werden. Im übrigen läuft das Programm wie folgt ab:

Zeile 20 Umschaltung auf 80 Zeichen pro Zeile

Zeilen 30
bis 260 Eingabe des Gleichungssystems Besonderheit hier:
Zeile 40 Das Feld wird für die einzelnen Faktoren erst dimensioniert, nachdem die Anzahl der Unbekannten eingegeben worden ist. Damit geht einerseits nicht zuviel Speicherplatz verloren, andererseits lassen sich aber Gleichungen mit beliebig vielen Unbekannten lösen.

Zeile 170 Die vorherige Zeile wird gelöscht, der übrige Text bleibt stehen.

Zeilen 280
bis 370 Lösung des Gleichungssystems
Am Ende ist in jeder Gleichung neben dem Restglied nur noch ein Glied vorhanden, und zwar in der 1. Gleichung das 1. Glied, in der 2. Gleichung das 2. Glied usw.
Zeile 420 Berechnung der jeweiligen Unbekannten durch Division des Restgliedes durch das jeweilige noch vorhandene Glied

Benötigter Speicherplatz: etwa 1,6 K.

```
10 REM Programm lin 2.0
20 MODE 2
30 PRINT "Anzahl der Unbekannten :";:INPUT n
40 DIM a(n,n+1),x(n)
50 CLS
60 PRINT "Geben Sie die Faktoren der einzelnen    Gl
eichungen ein."
70 PRINT:PRINT"Geben Sie alle Werte einer Gleichung
einschliesslich des Restgliedes ein, bevor Sie zur n
aechsten uebergehen."
80 PRINT:PRINT"Beruecksichtigen Sie fehlende Unbekan
nte in einer Gleichung durch Eingabe von 0."
90 LOCATE 1,25:PRINT"Weiter mit beliebiger Taste"
100 IF INKEY$="" THEN 100
110 LOCATE 1,25:PRINT SPC(27)
120 LOCATE 1,12 : PRINT"A 1,3 = 1.Gleichung, 3.Glied
 etc"
130 PRINT:PRINT"Restglied =  r e c h t s  vom Gleich
heitszeichen !!!!!"
140 FOR i = 1 TO n
150 FOR k = 1 TO n
160 LOCATE 5,18 : PRINT "A";i;",";k;:INPUT a(i,k)
170 LOCATE 5,18 : PRINT SPC(30)
180 NEXT k
190 LOCATE 5,18 : PRINT "B";i;:INPUT a(i,n+1)
200 LOCATE 5,18 : PRINT SPC(30)
210 NEXT i
220 PRINT:PRINT:PRINT "War die Eingabe i.o. ( /n) ";
:INPUT a$
230 IF a$ = "" THEN 260
240 IF a$ = "n" THEN 50
```

```
250 GOTO 220
260 CLS
270 PRINT"Ich arbeite. Bitte einen Augenblick Geduld
"
280 FOR k = 1 TO n
290 FOR i = 1 TO n
300 IF i=k THEN GOTO 360
310 f= -a(i,k)/a(k,k)
320 FOR l=1 TO n+1
330 b=a(k,l) * f
340  a(i,l) = a(i,l) + b
350 NEXT l
360 NEXT i
370 NEXT k
380 CLS
390 PRINT "We've got it !!!"
400 PRINT
410 FOR i=1 TO n
420 x(i) = a(i,n+1)/a(i,i)
430 PRINT "x";i;" = ";x(i)
440 NEXT i
450 LOCATE 1,23 : PRINT"Soll noch ein weiteres Gleic
hungssystem geloest werden ( /n) ";:INPUT a$
460 IF a$ = "" THEN RUN 20
470 IF a$ = "n" THEN 490
480 GOTO 450
490 MODE 1
500 LOCATE 3,10 : PRINT"Schoenen Feierabend" : LOCAT
E 1,24
510 END
```

8.2 Determinantenverfahren

Im Gegensatz zu dem vorstehend beschriebenen Programm erfolgt hier eine Prüfung auf Lösbarkeit des Systems. Dazu wird zunächst die Koeffizientendeterminante D des Gleichungssystems berechnet, d.h. die Determinante, deren einzelne Glieder aus den Faktoren a_{11} bis a_{nm} bestehen. Lösbar ist das System, wenn $D\langle\rangle 0$ ist.

Daß D zu 0 wird, kann verschiedene Ursachen haben; diese werden nicht näher untersucht.

Die eigentliche Lösung erfolgt jetzt nach der Cramerschen Regel. Hierzu wird nacheinander jeweils eine Spalte der Koeffizientendeterminante ersetzt durch die Faktoren b_1 bis b_n und dann diese neue Determinante durch D geteilt; der Quotient stellt die jeweilige Gleichungslösung dar.

Beispiel

Im Folgenden wird das in Abschnitt 8.1 angegebene Beispiel erneut durchgerechnet. Für die Berechnung des Eigenwertes der Determinanten sei auf Kapitel 6 verwiesen.

Gleichung I: $2x_1 + x_2 - x_3 = -2$

Gleichung II: $x_1 - 3x_2 - 2x_3 = -5$

Gleichung III: $-x_1 - 4x_2 + x_3 = 7$

$$D = \begin{vmatrix} 2 & 1 & -1 \\ 1 & -3 & -2 \\ -1 & -4 & 1 \end{vmatrix} = -14\langle\rangle 0$$ es gibt also eine eindeutige Lösung

$$D_1 = \begin{vmatrix} -2 & 1 & -1 \\ -5 & -3 & -2 \\ 7 & -4 & 1 \end{vmatrix} = 28 \qquad x_1 = D_1/D = \frac{-28}{-14} = 2$$

$$D_2 = \begin{vmatrix} 2 & -2 & -1 \\ 1 & -5 & -2 \\ -1 & 7 & 1 \end{vmatrix} = 14 \qquad x_2 = D_2/D = \frac{14}{-14} = -1$$

$$D_3 = \begin{vmatrix} 2 & 1 & -2 \\ 1 & -3 & -5 \\ -1 & -4 & 7 \end{vmatrix} = -70 \qquad x_3 = D_3/D = \frac{-70}{14} = 5$$

Beide Verfahren kommen also zu der gleichen Lösung.

Literaturhinweise

G: [5], S. 264 bis 267; G: [6], S. 76 ff; G: [8], S. 134 bis 137; G: [25], S. 58 ff; G, W: [1], S. 75 ff; G, W: [9], S. 71 bis 81; G, W: [22], S. 72 ff.

Programmbeschreibung

Für die Eingabe des Gleichungssystems gilt das bereits in Abschnitt 8.1 Gesagte.

Zeile 20
bzw.
Zeile 670
bis 910 Eingabe des Gleichungssystems.
Anzahl der Gleichungen = *Uzahl*
Abspeicherung im Feld *gs*

Zeile 40 Dimensionierung der Determinante *d*, deren Glieder im Laufe der nachfolgenden Berechnungen unterschiedliche Werte annehmen.

Zeile 50
bis 90 Zunächst werden die Faktoren der Koeffizientendeterminante nach *d* gespeichert.

Zeile 100
bzw.
Zeile 520
bis 660 Eigenwertbestimmung von *d*.
Der Eigenwert liegt anschließend in der Variablen *dew* vor.

Zeile 120
bzw.
Zeile 320
bis 510 Eigenes Ausgabeprogramm für den Fall, daß die Koeffizientendeterminante gleich 0 ist, da das Gleichungssystem nicht lösbar ist. Die Werte der einzelnen Zählerdeterminanten werden ausgegeben.

Zeile 130
bzw.
Zeile 930
bis 1080 Berechnung der Zählerdeterminante für den «Normalfall». Abspeicherung im Feld $x(i)$.

Zeilen 150
bis 170 Berechnung der einzelnen Unbekannten

Zeilen 180
bis 230 Lösungsausgabe

Benötigter Speicherplatz: etwa 3 K.

```
10 REM Programm lin det
20 GOSUB 670 : REM Eingabe des Gleichungssystems
30 danz = Uzahl
40 DIM d(danz,danz)
50 FOR k = 1 TO danz : REM Umspeicherung der Koeffiz
ientendeterminante
60 FOR i = 1 TO danz
70 d(k,i) = gs(k,i)
80 NEXT i
90 NEXT k
100 GOSUB 520 : REM Eigenwertbestimmung
110 kd = dew
```

```
120 IF kd = 0 THEN 320 : REM eigenes Ausgabeprogramm
130 GOSUB 930 : REM Berechnung der einzelnen Zaehler
determinanten
140 REM Berechnung der einzelnen Zaehlerdeterminante
n
150 FOR i = 1 TO uzahl
160 x(i) = x(i) / kd
170 NEXT i
180 REM Loesungsausgabe
190 CLS
200 PRINT"So,jetzt haben wir's geschafft!" : PRINT
210 FOR i = 1 TO uzahl
220 PRINT"x";i;" = ";x(i)
230 NEXT i
240 PRINT:PRINT:PRINT"Soll noch ein Gleichungssystem
 geloest werden ( /n) ";:INPUT a$
250 IF a$ = "" THEN RUN
260 IF a$ = "n" THEN 280
270 GOTO 240
280 CLS : LOCATE 15,10 : PRINT"Na gut. Wer nicht wil
l, der hat schon."
290 LOCATE 30,13 : PRINT"Tschuess"
300 LOCATE 1,23
310 END
320 REM Ausgabe fuer Koeffizientendeterminante = 0
330 CLS
340 PRINT"Koeffizientendeterminante = 0 "
350 PRINT
360 PRINT"Moegliche Ursachen"
370 PRINT" - lineare Abhaengigkeit von mindestens 2
Gleichungen"
```

```
380 PRINT "  (wenn mindestens eine Zaehlerdeterminan
te = 0)"
390 PRINT
400 PRINT " - Widerspruch im Gleichungssystem"
410 PRINT "   (wenn alle Zaehlerdeterminanten <> 0)"
420 PRINT
430 PRINT" - homogenes Gleichungssystem"
440 GOSUB 930 : REM Berechnung der einzelnen Zaehler
determinanten
450 PRINT:PRINT:PRINT
460 PRINT "Werte der einzelnen Zaehlerdeterminanten"
470 PRINT
480 FOR i = 1 TO uzahl
490 PRINT x(i)
500 NEXT i
510 GOTO 240
520 REM UP Eigenwertbestimmung von Determinanten
530 FOR m = 1 TO danz
540 FOR k = 1 TO danz
550 IF k = m THEN GOTO 600
560 a1 = -d(k,m)/d(m,m)
570 FOR i = 1 TO danz
580 d(k,i) = d(k,i) + d(m,i)*a1
590 NEXT i
600 NEXT k
610 NEXT m
620 dew = 1
630 FOR i = 1 TO danz
640 dew = dew*d(i,i)
650 NEXT i
660 RETURN
```

```
670 REM Eingabe des Gleichungssystems
680 CLS
690 PRINT"Wieviele Unbekannte hat das Gleichungssyst
em ";
700 INPUT uzahl
710 DIM gs(uzahl,uzahl+1),x(uzahl)
720 PRINT
730 PRINT"A3 = Faktor vor x3, A2 = Faktor vor x2 etc
."
740 PRINT
750 PRINT"fehlende Glieder durch 0 beruecksichtigen
!!!"
760 FOR k = 1 TO uzahl
770 LOCATE 1,10 : PRINT k;"te Gleichung"
780 FOR i = 1 TO uzahl
790 LOCATE 1,12 : PRINT"A";i;" : ";
800 INPUT gs(k,i)
810 LOCATE 1,12 : PRINT"                        "
820 NEXT i
830 LOCATE 1,12 :PRINT "Y : ";
840 INPUT gs(k,uzahl+1)
850 NEXT k
860 PRINT
870 PRINT:PRINT " Eingabe o.k. ( /n) ";: INPUT a$

880 IF a$ = "" THEN 910
890 IF a$ = "n" THEN RUN
900 GOTO 870
910 PRINT
920 RETURN
930 REM UP zaehlerdeterminantenberechnung
```

```
940 FOR n = 1 TO uzahl
950 FOR k = 1 TO danz
960 IF k = n THEN GOTO 1000
970 FOR i = 1 TO danz
980 LET d(i,k) = gs (i,k)
990 NEXT i
1000 NEXT k
1010 FOR i = 1 TO danz
1020 d(i,n) = gs(i,danz+1)
1030 NEXT i
1040 GOSUB 530
1050 REM Eigenwertbestimmung
1060 x(n) = dew
1070 NEXT n
1080 RETURN
```

9
Fehlerrechnung

Wenn der Wert einer Größe gemessen wird, ist dieser Meßwert normalerweise mit Fehlern behaftet. Man unterscheidet hier systematische Fehler wie Fehler der Analysenmethode usw. von zufälligen Fehlern. Systematische Fehler sind grundsätzlich feststellbar und können demzufolge berücksichtigt werden, zufällige Fehler können jedoch nur mit Hilfe der Wahrscheinlichkeitsrechnung erfaßt und korrigiert werden.

Soll eine Größe x bestimmt werden, wird man normalerweise die Einzelmessung n-mal wiederholen. Der wahrscheinliche Wert der Größe ist dann der Mittelwert.

$$\bar{x} = \frac{x_1 + x_2 + \ldots + x_n}{n} = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{9.1}$$

$\bar{x}$ ist ein Schätzwert für den wahren Wert von x. Daß für die Bildung von $\bar{x}$ das arithmetische Mittel und nicht z.B. das geometrische herangezogen werden muß, kann über die Methode der minimalen Fehlerquadrate nachgewiesen werden, die von Gauß zum erstenmal zur Bahnberechnung von Planeten angewendet wurde.

v_i bezeichne die jeweilige Abweichung der Meßwerte x_i vom tatsächlichen Wert x. Die Forderung nach Gauß ist dann, daß die Summe der Fehlerquadrate zu einem Minimum werden soll:

$$\sum_{i=1}^{n} v_i^2 = \text{Minimum!}$$

bzw. in der Schreibweise nach Gauß

$$[vv] = \text{Minimum!}$$

Der quadratische Fehler wird genommen, damit sich positive und negative Fehler nicht aufheben; außerdem werden «Ausreißer» verstärkt berücksichtigt.

Ersetzt man v_i durch $x - x_i$, so ergibt sich:

$$[vv] = (\bar{x} - x_1)^2 + (\bar{x} - x_2)^2 + \ldots + (\bar{x} - x_n)^2$$

Daß diese Summe zu einem Minimum werden soll, heißt, daß die erste Ableitung zu 0 werden muß (vgl. Kapitel 2):

$$\frac{d[vv]}{dx} = 2(\bar{x} - x_1) + 2(\bar{x} - x_2) + \ldots + 2(\bar{x} - x_n) = 0$$

$$\bar{x} - x_1 + \bar{x} - x_2 + \ldots + \bar{x} - x_n = 0$$

$$x_1 + x_2 + \ldots + x_n = n\bar{x}$$

$$\frac{x_1 + x_2 + \ldots + x_n}{n} = \bar{x} = \text{arithm. Mittel}$$

Es reicht jedoch häufig nicht aus, den Mittelwert $\bar{x}$ zu kennen, sondern man muß ein Maß dafür haben, wie wahrscheinlich dieser Wert ist bzw. in welcher Bandbreite der wahre Wert x zu suchen ist. Hierfür kennt man zunächst die Varianz s^2, die folgendermaßen definiert ist:

$$s^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_1 - x)^2 \qquad (9.2)$$

Die positive Wurzel aus s^2 ist die Standardabweichung s, auch mittlerer Fehler der Einzelmessung oder empirische Streuung genannt.

$$s = \sqrt{\frac{\sum_{i=1}^{n} (x_i - x)^2}{n-1}} \qquad (9.3)$$

Wenn der Meßwert angegeben wird, lautet die Angabe dann:

$$x \pm s$$

wobei s sowohl in Prozent als auch in absoluten Einheiten angegeben werden kann.

Da die Einzelmessungen mit Fehlern behaftet sind, muß auch der Mittelwert mit Fehlern behaftet sein. Ein Maß hierfür ist die Streuung des Mittelwertes; dieser Wert ist auch als Standardabweichung des Mittelwertes bekannt:

$$s_x = \sqrt{\frac{\sum_{i=1}^{n} (x_i - x)^2}{n(n-1)}} \qquad (9.4)$$

Beispiel

Die Masse eines Produktgefäßes wurde 8mal überprüft:

1	1	2	3	4	5	6	7	8
x [kg]	20,34	20,34	20,38	20,20	20,31	20,50	20,33	20,34

Wie groß ist der Mittelwert und die Standardabweichung?

$$\bar{x} = \frac{20{,}34 + 20{,}34 + 20{,}38 + 20{,}20 + 20{,}31 + 20{,}50 + 20{,}33 + 20{,}39}{8}$$

$$= 20{,}35\ \text{kg}$$

i	1	2	3	4	5	6	7	8
$x - x_i$	0,01	0,01	−0,03	0,15	0,04	−0,15	0,02	−0.04

$$S = \sqrt{\frac{0{,}01^2 + 0{,}01^2 + 0{,}03^2 + 0{,}15^2 + 0{,}04^2 + 0{,}15^2 + 0{,}02^2 + 0{,}04^2}{8 - 1}}$$

$S = 0{,}08 = 0{,}41\,\%$

$\bar{x} = 20{,}35\ \text{kg} \pm 0{,}41\,\%$

Literaturhinweise

G, W: [2], S. 41 bis 49; G, W: [8]; G, W: [10], S. 188 bis 191; G, W: [14]; G, W: [17], S. 100 ff.

Programmbeschreibung

	Das Programm Fehler kann bis zu 100 Meßwerte verarbeiten; ist die Anzahl höher, muß x in
Zeile 20	entsprechend höher dimensioniert werden. In den
Zeilen 50	
bis 110	werden die Meßwerte eingegeben und die Anzahl festgestellt; beendet wird der Vorgang durch Eingabe von 333.33. In
Zeile 130	
bis 180	wird der Mittelwert berechnet, in
Zeile 190	

bis	240	die empirische Varianz und in
Zeile	260	als Wurzel aus der Varianz die empirische Streuung. Schließlich wird in
Zeile	270	
bis	290	noch die Standardabweichung berechnet, und zwar sowohl in absoluten Einheiten als auch in %. Über die
Zeile	300	
bis	360	werden alle berechneten Werte schließlich ausgegeben.

Programmlänge: etwa 1,7 K.

```
10 REM Programm Fehler
20 DIM x(100)
30 REM Dateneingabe
40 CLS
50 PRINT"Geben Sie die einzelen Messwerte ein. Beend
en Sie die Eingabe durch 333.33."
60 WINDOW #1,1,80,3,25
70 FOR i=1 TO 100
80 PRINT#1,"x";i;": ";:INPUT#1,x(i)
90 IF x(i)=333.33 THEN 110
100 NEXT i
110 anzahl = i-1
120 CLS
130 REM Mittelwert
140 sum=0
150 FOR i=1 TO anzahl
160 sum=sum + x(i)
170 NEXT i
180 mw=sum/anzahl
190 REM Empirische Varianz
200 sf2=0
```

```
210 FOR i=1 TO anzahl
220 sf2=sf2 + (mw-x(i))*(mw-x(i))
230 NEXT i
240 var=sf2/(anzahl-1)
250 REM Empirische Streuung
260 str=SQR(var)
270 REM Standardabweichungen
280 stabw=SQR(sf2/(anzahl*(anzahl-1)))
290 stabwrel=stabw/ABS(mw)*100
300 REM Ausgabe
310 CLS
320 PRINT"Mittelwert             :    ";mw
330 PRINT:PRINT"Standardabweichung :  +-";stabw
340 PRINT"                         +-";stabwrel;" %"
350 PRINT:PRINT"Empir. Streuung    :    ";str
360 PRINT:PRINT"Empir. Varianz     :    ";var
370 LOCATE 1,23
380 END
```

10
Ausgleichsrechnung

10.1 Lineare Regression

Grundlagen

Ein oft auftauchendes Problem in nahezu allen Bereichen der Wirtschaft besteht darin, zu einer gegebenen Reihe von Meßwertpaaren den funktionalen Zusammenhang zu finden, der die Werte optimal beschreibt, gleichzeitig aber Meßfehler möglichst eliminiert. Normalerweise geht man so vor, daß man einen bestimmten Funktionstyp auswählt, anhand der Meßwerte für diesen Typ entsprechende Parameter berechnet und dann schließlich mittels eines speziellen Kriteriums die Güte der Annäherung beurteilt. Führt man dies für verschiedene Funktionstypen durch, kann man anhand der Gütekriterien feststellen, welcher Funktionstyp am besten geeignet ist.

Die einfachste Funktion, die für diese Ausgleichsrechnungen betrachtet wird, ist die Gerade.

$$y = a_0 + a_1 x \tag{10.1}$$

In diesem Fall spricht man von linearer Regression. Hier geht es darum, die Faktoren a_0 und a_1 aus Gleichung (10.1) zu bestimmen. Hierfür gibt es diverse Methoden; am bekanntesten ist die Approximation durch orthogonale Funktionssysteme, vorgeschlagen von Fourier und Tschebyscheff [3] (Seiten 526 bis 533) und die Methode der kleinsten Fehlerquadrate nach Gauß. Im Folgenden wird wie auch in Kapitel 9 das Gaußsche Verfahren verwendet.

Die zentrale Forderung nach Gauß lautet, daß die Summe der Fehlerquadrate zu einem Minimum werden soll (vgl. Kapitel 9). Als Fehler v_i ist hierbei jeweils der Abstand des Meßwertes y_i zu dem berechneten Funktionswert $f(x_i)$ zu verstehen, die beide zu dem Meßwert x_i gehören.

$$\sum_{i=1}^{n} (y_i - f(x_i))^2 = [vv] = \text{Minimum}$$

$$[vv] = (a_1 x_1 + a_0 - y_1)^2 + (a_1 x_2 + a_0 - y_2)^2 + \ldots$$
$$+ (a_1 x_1 + a_0 - y_1)^2 \qquad (10.2)$$

Aus dieser Gleichung müssen a_1 und a_0 so bestimmt werden, daß $[vv]$ zu einem Minimum wird. Die 1. Ableitung muß also zu 0 werden; da a_1 und a_0 beides Veränderliche sind, muß nach diesen beiden Größen partiell abgeleitet werden. x_i und y_i sind als Konstanten zu betrachten (für den Begriff der partiellen Ableitung und für die verwendeten Ableitungsregeln siehe z.B. [22], S. 252 ff).

$$\frac{d[vv]}{da_0} = 2(a_1 x_1 + a_0 - y_1) + 2(a_1 x_2 + a_0 - y_2) + \ldots$$
$$+ 2(a_1 x_n + a_0 - y_1) = 0 \qquad (10.3)$$

$$\frac{d[vv]}{da_1} = 2(a_1 x_1 + a_0 - y_1) x_1 + 2(a_1 x_2 + a_0 - y_2) x_2$$
$$+ \ldots + 2(a_1 x_n + a_0 - y_n) x_n = 0 \qquad (10.4)$$

Aus Gleichung (10.3) folgt nach Umformung die Berechnungsgleichung für a_0

$$a_1(x_1 + x_2 + \ldots + x_1) + n * a_0 = y_1 + y_2 + \ldots + y_1$$

$$a_0 = \frac{1}{n} \sum_{i=1}^{n} y_1 - \frac{a_1}{n} \sum_{i=1}^{n} x_i \qquad (10.5)$$

Im Folgenden werden aus Vereinfachungsgründen die Laufwerte für die Summationen weggelassen; alle Summen laufen von 1 bis n.
Aus Gleichung (10.4) folgt nach Umformung:

$$a_1 x_1^2 + a_0 x_1 - y_1 x_1 + a_1 x_2^2 + a_0 x_2 - y_2 x_2$$
$$+ \ldots + a_1 x_n^2 + a_0 x_n - y_n x_n = 0$$

$$a_1 \sum x_i^2 + a_0 \sum x_i = \sum x_i y_i$$

Wenn man in diese Gleichung die Gleichung (10.5) einsetzt, ergibt sich die Berechnungsformel für a_1:

$$a_1 \sum x_i^2 + \frac{\sum y_i - a_1 \sum x_i}{n} \sum x_i = \sum x_i y_i$$

$$a_1 \sum x_i^2 + \frac{\sum y_i \sum x_i}{n} - \frac{a_1 (\sum x_i)^2}{n} = \sum x_i y_i$$

$$a_1 \left(\sum x_i^2 - \frac{(\sum x_i)^2}{n} \right) = \sum x_i y_i - \frac{\sum y_i \sum x_i}{n}$$

$$a_1 = \frac{\sum x_i y_i - \frac{\sum x_i \sum y_i}{n}}{\sum x_i^2 - \frac{(\sum x_i)^2}{n}} \tag{10.6}$$

Als Gütekriterium wird häufig das Bestimmtheitsmaß R^2 verwendet [11], (S. 579 bis 591). Zu dessen Berechnung wird zunächst der Mittelwert $\bar{y}$ der y-Werte gebildet und dann die Summe der quadratischen Abweichungen der Meßwerte y_i vom Mittelwert und der berechneten Funktionswerte $f(x_i)$ vom Mittelwert ermittelt.

$$SQMY = \sum (f(x_i) - \bar{y})^2 \tag{10.7}$$

$$SQAY = \sum (y - \bar{y})^2 \tag{10.8}$$

Das Bestimmtheitsmaß ist definiert als der Quotient dieser beiden Summen.

$$R^2 = \frac{SQMY}{SQAY} \tag{10.9}$$

Für den Idealfall, daß die Ausgleichsgerade durch alle Meßpunkte geht, sind beide Summen gleich, R^2 ist demnach gleich 1. Je mehr R^2 von 1 verschieden ist, desto schlechter ist die Annäherung.

Eine weitere Möglichkeit für ein Gütekriterium ist die Bestimmung der Abweichung *dy*. Hier wird die quadratische Summe der Differenzen zwischen Meßwert y_i und Funktionswert $f(x_i)$ gebildet, diese Summe auf die Anzahl der Meßwerte bezogen und hieraus die Wurzel bestimmt.

$$dy = \sqrt{\frac{\sum (y_i - f(x_i))^2}{n}} \tag{10.10}$$

Wird dieser Wert auf den Mittelwert bezogen, erhält man die relative Abweichung *dyrel*

$$dyrel = \frac{dy}{\bar{y}} * 100\,\% \qquad (10.11)$$

dy und *dyrel* sind also mit den in Kapitel 9 besprochenen Standardabweichungen verwandt.

Beispiel

Für folgende Wertetabelle ist die optimale Gerade zu bestimmen. Des weiteren sind die Gütekriterien zu berechnen.

i	1	2	3	4	5	6
x	0,5	1	2	3	5	7
y	0,1	0,3	0,7	1,5	2	4

$$\sum x_i = 18{,}5 \qquad \sum y_i = 8{,}6 \qquad \sum x_1 y_i = 44{,}25 \qquad \sum x_i^2 = 88{,}25$$

$$a_1 = \frac{44{,}25 - 18{,}5 \cdot \dfrac{8{,}6}{6}}{88{,}25 - \dfrac{18{,}5^2}{6}} = 0{,}568$$

Hiermit folgt aus Gleichung (10.5):

$$a_0 = \frac{8{,}6}{6} - \frac{0{,}568}{6} \cdot 18{,}5 = -0{,}318$$

Die Gleichung der optimalen Geraden lautet also

$$y = -0{,}318 + 0{,}568x$$

Mittelwert: $\bar{y} = 8{,}6/6 = 1{,}433$

1	1	2	3	4	5	6
$f(x_i)$	−0,035	0,250	0,818	1,386	2,522	3,659
$f(x_i) - \bar{y}$	−1,468	−1,183	−0,615	−0,047	1,089	2,226
$y_i - \bar{y}$	−1,333	−1,133	−0,733	0,067	0,567	2,567
$y_i - f(x_i)$	0,135	0,050	−0,118	0,114	−0,522	−0,341

$$SQMY = \sum(f(x_i - \bar{y})^2 = 10{,}08$$

$$SQAY = \sum(y_i - \bar{y})^2 = 10{,}51$$

$$R^2 = 10{,}08/10{,}51 = 0{,}959$$

$$\sum(y_i - f(x_i))^2 = 0{,}439$$

$$dy = \sqrt{\frac{0{,}439}{6}} = 0{,}270$$

$$dyrel = \frac{0{,}0266}{1{,}433} \cdot 100\,\% = 18{,}9\,\%$$

Das Programm zur linearen Regression ist unter Abschnitt 10.2 zu finden, da es mit den dortigen Programmen verknüpft wurde.

Literaturhinweise

G, W: [14], S. 80 ff; G, W: [18]; G, W: [22], S. 340 bis 343.

10.2 Anpassung durch einfache Funktionen

Grundlagen

Wenn die Meßwertanpassung durch eine Gerade kein befriedigendes Ergebnis liefert, wird man normalerweise zunächst auf andere einfache Funktionen zurückgreifen; hier sind insbesondere die Logarithmusfunktion, die e-Funktion und die Potenzfunktion zu nennen. Man geht dann so vor, daß man die jeweilige Funktion durch Termersetzung in eine Geradengleichung überführt und die zu berechnenden Koeffizienten über Gleichung (10.5) und (10.6) ermittelt. Da durch die Termersetzung neue Variablen eingeführt werden, lautet die Geradengleichung

$$p = a_0 + a_1 z \qquad (10.12)$$

wenn in der Ausgangsgleichung die Variablen x und y heißen sollen.

Die Faktoren a_0 und a_1 berechnen sich also nach

$$a_1 = \frac{\sum z_i p_i - \frac{\sum z_i \sum p_i}{n}}{\sum z_i^2 - \frac{(\sum z_i)^2}{n}} \qquad (10.13)$$

$$a_0 = \frac{\sum p_i}{n} - \frac{a_1 \sum z_i}{n} \qquad (10.14)$$

Die Gleichungen für die Gütekriterien (10.7) bis (10.11) können dagegen unverändert übernommen werden.

Logarithmusfunktion

Funktionsgleichung: $\qquad y = a_0 + a_1 \ln(x) \qquad (10.15)$

Ersetzt man y durch p und $\ln(x)$ durch z, ergibt sich die Geradengleichung (10.12), so daß in Gleichung (10.13) und (10.14) eingesetzt werden kann. Beim Einsetzen wird die Termersetzung wieder rückgängig gemacht, d. h., in beiden Gleichungen wird p durch y und z durch $\ln(x)$ ersetzt.

Somit ergibt sich:

$$a_1 = \frac{\sum y_i \ln(x_i) - \frac{\sum \ln(x_i) \sum y_i}{n}}{\sum (\ln(x_i))^2 - \frac{(\sum \ln(x_i))^2}{n}} \tag{10.16}$$

$$a_0 = \frac{\sum y_i}{n} - \frac{a_1 \ln(x_i)}{n} \tag{10.17}$$

Exponentialfunktion

Funktionsgleichung: $y = a_0\, \mathrm{e}^{a_1 x}$ (10.18)

Diese Gleichung wird zunächst beidseitig logarithmiert:

$$\ln(y) = \ln(a_0) + a_1 x \tag{10.19}$$

Hier wird jetzt $\ln(y)$ durch p und x durch z ersetzt. Beim Einsetzen in Gleichung (10.14) muß statt a_0 zunächst $\ln(a_0)$ geschrieben werden, da Gleichung (10.19) sonst keine Geradengleichung darstellt. Beim Einsetzen wird die Termersetzung wieder rückgängig gemacht:

$$a_1 = \frac{\sum x_i \ln(y_i) - \frac{\sum x_1 \sum \ln(y_i)}{n}}{\sum x_i^2 - \frac{(\sum x_i)^2}{n}} \tag{10.20}$$

$$\ln(a_0) = \frac{\sum \ln(y_i)}{n} - \frac{a_1 \sum x_i}{n} \quad \text{bzw.}$$

$$a_0 = \exp\left(\frac{\sum \ln(y_i)}{n} - \frac{a_i \sum x_i}{n}\right) \tag{10.21}$$

Potenzfunktion

Funktionsgleichung: $y = a_0\, x^{a_1}$ (10.22)

Logarithmieren führt zu

$$\ln(y) = \ln(a_0) + a_1 \ln(x) \tag{10.23}$$

Termersetzung: $\ln(y) = p \qquad \ln(x) = z$

Auch hier muß in Gleichung (10.14) zunächst $\ln(a_0)$ statt a_0 geschrieben werden.

$$a_1 = \frac{\sum(\ln(x_i))(\ln(y_i)) - \dfrac{\sum x_i \sum \ln(y_i)}{n}}{\sum x_i^2 - \dfrac{\sum x_i)^2}{n}} \tag{10.24}$$

$$\ln(a_0) = \frac{\sum \ln(y_i)}{n} - \frac{a_1 \sum x_i}{n} \qquad \text{bzw.}$$

$$a_0 = \exp\left(\frac{\sum \ln(y_i)}{n} - \frac{a_1 \sum x_i}{n}\right) \tag{10.25}$$

Beispiele

Im Folgenden werden die Meßwerte aus Abschnitt 10.1 durch die genannten Funktionstypen angenähert und die dazugehörigen Gütekriterien berechnet.

Logarithmusfunktion

$$\sum y_i \ln(x_i) = 13{,}07 \qquad \sum \ln(x_i) = 4{,}653$$

$$\sum y_i = 8{,}6 \qquad \sum(\ln(x_i))^2 = 8{,}545$$

$$\bar{y} = (\sum y_i)/n = 1{,}433$$

Gleichung (10.16): $$a_1 = \frac{13{,}07 - \dfrac{4{,}653 \cdot 8{,}6}{6}}{8{,}545 - \dfrac{4{,}653^2}{6}} = 1{,}297$$

Gleichung (10.17): $$a_0 = \frac{8{,}6}{6} - \frac{1{,}297}{6} \cdot 4{,}653 = 0{,}428$$

$$SQMY = 8{,}30 \qquad SQAY = 10{,}51$$

Gleichung (10.9): $$R^2 = \frac{8{,}30}{10{,}51} = 0{,}79$$

$$\sum(y_i - f(x_i))^2 = 2{,}225$$

Gleichung (10.10): $$dy = \sqrt{\frac{2{,}225}{6}} = 0{,}609$$

Gleichung (10.11): $$dyrel = \frac{0{,}609}{1{,}433} \cdot 100\,\% = 42{,}5\,\%$$

Die Gleichung lautet also $y = 0{,}428 + 1{,}297 \ln(x)$, die Anpassung ist jedoch völlig unzureichend.

Exponentialfunktion

$\sum x_i = 18{,}5$	$\sum \ln(y_i) = -1{,}378$	$\sum x_i \ln(y_i) = 11{,}32$
$\sum x_i^2 = 88{,}25$	$\sum y_i = 8{,}6$	$\bar{y} = 1{,}433$
$SQMY = 22{,}01$	$SQAY = 10{,}51$	$\sum (y_i - f(x_i))^2 = 3{,}202$

Gleichung (10.20): $$a_1 = \frac{11{,}32 - \frac{18{,}5\,(-1{,}378)}{6}}{88{,}25 - \frac{18{,}5^2}{6}} = 0{,}499$$

Gleichung (10.21): $$a_0 = \exp\left(\frac{-1{,}378}{6} - 0{,}499\,\frac{18{,}5}{6}\right) = 0{,}171$$

Gleichung (10.9): $$R^2 = \frac{22{,}01}{10{,}51} = 2{,}09$$

Gleichung (10.10): $$dy = \sqrt{\frac{3{,}202}{6}} = 0{,}73$$

Gleichung (10.11): $$dyrel = \frac{0{,}73}{1{,}433} \cdot 100\,\% = 51\,\%$$

Die Gleichung $y = 0{,}171 \exp(0{,}499\,x)$ ist also schon besser als die vorher berechnete, aber immer noch wesentlich schlechter als die lineare Regression.

Potenzfunktion

$$\sum(\ln(x_i))(\ln(y_i)) = 5{,}607 \qquad \sum \ln(y_i) = -1{,}378$$

$$\sum \ln(x_i) = 4{,}654 \qquad \sum(\ln(x_i))^2 = 8{,}545$$

$$\sum y_i = 8{,}6 \qquad \bar{y} = 1{,}433$$

Gleichung (10.24): $$a_1 = \frac{5{,}607 - \dfrac{-1{,}378 \cdot 4{,}654}{6}}{8{,}545 - \dfrac{4{,}654^2}{6}} = 1{,}353$$

Gleichung (10.25): $$a_0 = \exp\left(\frac{-1{,}378}{6} - 1{,}353\,\frac{4{,}654}{6}\right) = 0{,}278$$

$$SQMY = 10{,}62 \qquad SQAY = 10{,}52$$

Gleichung (10.9): $$R^2 = \frac{10{,}62}{10{,}52} = 1{,}01$$

$$\sum(y_i - f(x_i))^2 = 0{,}296$$

Gleichung (10.10): $$dy = \sqrt{\frac{0{,}296}{6}} = 0{,}22$$

Gleichung (10.11): $$dyrel = \frac{0{,}22}{1{,}433} \cdot 100\,\% = 15{,}5\,\%$$

Funktionsgleichung: $$y = 0{,}278\,x^{1{,}353}$$

Mit der Anpassung durch eine Potenzfunktion erreicht man also schon eine fast optimale Annäherung an die Meßwerte.

Literaturhinweise

G, W: [14], S. 80 ff; G, W: [18]; G, W: [22], S. 340 bis 343

Programmbeschreibung

Da mit dem Programm *Ausgleich* vier verschiedene Anpassungsmöglichkeiten gegeben sind, für die jeweils die Gütekriterien bestimmt werden, hat auch dieses Programm

	eine Länge von ca. 8 K. Nach Dimensionierung der Felder für die x- und y-Werte in
Zeile 30	deren Größe auf 100 festgesetzt ist, folgt in
Zeile 40 bis 130	ein Menü, über das die gewünschte Anpassungsart ausgewählt wird. Entsprechend der Codezahl verzweigt
Zeile 130	in die notwendigen Unterprogramme. Nach erfolgter Ausgleichung hat man über
Zeile 140 bis 250	zunächst die Möglichkeit, mit der erhaltenen Gleichung Zwischenwerte zu berechnen; diese werden einzeln auf dem Bildschirm ausgegeben. In den
Zeilen 260 bis 290	ist die Möglichkeit des Rücksprungs ins Menü gegeben, damit für die gleichen Meßwerte eine neue Anpassungsart gewählt werden kann. Diese Möglichkeit wird man nutzen, wenn die Gütekriterien aussagen, daß die Anpassung nicht gut genug war. Mittels des
Zeilen 300 bis 330	kann das Programm schließlich neu gestartet werden. In den
Zeilen 400 bis 540	ist das Unterprogramm für die lineare Regression abgelegt. Es benutzt wie alle anderen entsprechenden Programmteile die
Zeilen 1700 bis 1880	zur Eingabe der Meßwerte; in der Variablen*anzahl* wird die Zahl der Meßwerte festgehalten. Die Berechnung der Regressionsfaktoren erfolgt dann in den
Zeilen 430 bis 510	Die nächsten beiden Schritte sind dann wieder gleich für alle Anpassungsarten; nämlich die Benutzung der
Zeilen 2500 bis 2810	für die Berechnung der Gütekriterien und der
Zeilen 2000 bis 2100	für die Ausgabe der Ergebnisse. Prinzipiell der gleiche Vorgang läuft ab in den

Zeilen 600	
bis 820	für die Anpassung durch eine Exponentialfunktion, in den
Zeilen 1000	
bis 1220	für die Anpassung durch eine Logarithmusfunktion sowie in den
Zeilen 1300	
bis 1520	für die Anpassung durch eine Potenzfunktion. In allen drei Teilprogrammen kommt jedoch hinzu, daß vor Durchführung der Rechnungen geprüft werden muß, ob die gewählte Anpassung überhaupt möglich ist (s. Grundlagen). Wenn das nicht der Fall sein sollte, erfolgt die entsprechende Ausgabe und der Rücksprung ins Menü, damit eine andere Anpassungsart gewählt werden kann.

```
10 REM Programm Ausgleich
20 MODE 2
30 DIM x(100),y(100)
40 CLS : REM Menue
50 PRINT"Mit welchem Funktionstyp sollen die Messwer
te angepasst werden ?":PRINT"-----------------------
----------------------------
-----------------------------"
60 PRINT:PRINT"- Gerade (lineare Regression)      y =
 a + b*x";TAB(70);"(1)"
70 PRINT:PRINT"- Exponentialfunktion              y =
 a*exp(b*x)";TAB(70);"(2)"
80 PRINT:PRINT"- Logarithmusfunktion              y =
 a + b*log(x)";TAB(70);"(3)"
90 PRINT:PRINT"- Potenzfunktion                   y =
 a*x^b (a>0)";TAB(70);"(4)"
100 PRINT:PRINT"- mit gar keiner";TAB(70);"(5)"
110 INPUT was
120 IF was>5 OR was<1 THEN 40
```

```
130 ON was GOSUB 400,600,1000,1300,1600
140 CLS: PRINT"Soll die ermittelte Gleichung fuer In
terpolationen verwendet werden  ( /n)";:INPUT a$
150 IF a$="" THEN 170
160 IF a$="n" THEN 260 ELSE 140
170 PRINT"Beenden Sie die Interpolationsrechnungen d
urch Eingabe von x = 333.33"
180 PRINT:PRINT"x : ";:INPUT x
190 IF was = 1 THEN y = a + b*x
200 IF was = 2 THEN y = a*EXP(b*x)
210 IF was = 3 THEN y = a + b*LOG(x)
220 IF was = 4 THEN y = a*x^b
230 IF x = 333.33 THEN 260
240 PRINT"y = ";y
250 GOTO 180
260 weiter = 0
270 CLS : PRINT"Sollen die gleichen Messwerte fuer e
ine erneute Anpassung verwendet werden ( /n)":INPUT
a$
280 IF a$="" THEN weiter =1 : GOTO 40
290 IF a$="n" THEN 300 ELSE 190
300 CLS
310 PRINT"Soll eine weitere Anpassung mit anderen Me
sswerten durchgefuehrt werden ( /n)";:INPUT a$
320 IF a$="" THEN RUN
330 IF a$="n" THEN 1600 ELSE 300
340 END
400 REM UP/HP lineare Regression
410 a1$ = "Lineare Regression            y = a + b*x"
420 GOSUB 1700 : REM Eingabe der Messwerte
430 sum1=0 : sum2=0 : sum3=0 : sum4=0
```

```
440 FOR k=1 TO anzahl
450 sum1=sum1 + x(k)*y(k)
460 sum2=sum2 + x(k)
470 sum3=sum3 + y(k)
480 sum4=sum4 + x(k)*x(k)
490 NEXT k
500 b = (sum1 - (sum2*sum3/anzahl))/(sum4 - (sum2*su
m2/anzahl))
510 a = sum3/anzahl - (b*sum2/anzahl)
520 GOSUB 2500 : REM Guetekriterien
530 GOSUB 2000 : REM Ausgabe
540 RETURN
600 REM Anpssung durch Exponentialfunktion
610 GOSUB 1700 : REM Eingabe der Messwerte
620 a1$ = "Exponentialanpassung      y = a*exp(b*x)"

630 sum1=0 : sum2=0 : sum3=0 : sum4=0
640 FOR k=1 TO anzahl
650 IF y(k)<=0 THEN 760
660 sum1=sum1 + x(k)
670 sum2=sum2 + LOG(y(k))
680 sum3=sum3 + x(k)*LOG(y(k))
690 sum4=sum4 + x(k)*x(k)
700 NEXT k
710 b = (sum3 - (sum1*sum2/anzahl)) / (sum4 - (sum1*
sum1/anzahl))
720 a = EXP((sum2/anzahl) - (b*sum1/anzahl))
730 GOSUB 2500 : REM Guetekriterien
740 GOSUB 2000 : REM Ausgabe
750 RETURN
760 CLS
```

```
770 PRINT"Kurvenanpassungen ueber eine Exponentialfu
nktion sind nur moeglich, wenn alle   y(i) groesser
0 sind"
780 PRINT:PRINT"Hier ist aber y";k;" = ";y(k)
790 LOCATE 1,23
800 PRINT"Weiter mit beliebiger Taste"
810 IF INKEY$="" THEN 810
820 RETURN
1000 REM Kurvenanpassung per Logarithmusfunktion
1010 GOSUB 1700 : REM Messpunkteingabe
1020 a1$ = "Logarithmische Funktionsanpassung       y
 = a + b*log(x)
1030 sum1=0 : sum2=0 : sum3 = 0 : sum4=0
1040 FOR k=1 TO anzahl
1050 IF x(k)<= 0 THEN 1160
1060 sum1=sum1 + y(k)*LOG(x(k))
1070 sum2=sum2 + LOG(x(k))
1080 sum3=sum3 + LOG(x(k))*LOG(x(k))
1090 sum4=sum4 + y(k)
1100 NEXT k
1110 b = (sum1 - (sum2*sum4/anzahl)) / (sum3 - (sum2
*sum2/anzahl))
1120 a = (sum4 - (b*sum2))/anzahl
1130 GOSUB 2500 : REM Guetekriterien
1140 GOSUB 2000 : REM Ausgabe
1150 RETURN
1160 REM x<=0
1170 CLS
1180 PRINT"Eine logarithmische Kurvenanpassung ist n
icht moeglich, wenn ein x-Wert         kleiner oder
gleich 0 ist. Hier ist aber
```

```
x";k;" = ";x(k)
1190 LOCATE 1,23
1200 PRINT"Weiter mit beliebiger Taste"
1210 IF INKEY$="" THEN 1210
1220 RETURN
1300 REM Anpassung ueber eine Potenzfunktion
1310 a1$ = "Anpassung ueber die Potenzfunktion
 y = a*x^b"
1320 GOSUB 1700 : REM Messwerteingabe
1330 sum1=0 : sum2=0 : sum3=0 : sum4=0
1340 FOR k=1 TO anzahl
1350 IF x(k)<=0 OR y(k)<=0 THEN 1460
1360 sum1=sum1 + LOG(x(k))*LOG(y(k))
1370 sum2=sum2 + LOG(x(k))
1380 sum3=sum3 + LOG(y(k))
1390 sum4=sum4 + LOG(x(k))*LOG(x(k))
1400 NEXT k
1410 b = (sum1 - (sum2*sum3/anzahl)) / (sum4 - (sum2
*sum2/anzahl))
1420 a = EXP(sum3/anzahl  - (b*sum2/anzahl))
1430 GOSUB 2500 : REM Guetekriterien
1440 GOSUB 2000 : REM Ausgabe
1450 RETURN
1460 CLS
1470 PRINT"Bei einer Anpassung ueber eine Potenzfunk
tion muessen alle x(i) und alle y(i)   groesser 0 se
in."
1480 PRINT:PRINT"Hier ist aber x";k;" = ";x(k);"  un
d y";k;" = ";y(k)
1490 LOCATE 1,23
1500 PRINT"Weiter mit beliebiger Taste"
```

```
1510 IF INKEY$="" THEN 1510
1520 RETURN
1600 REM Schlussprogramm
1610 CLS
1620 LOCATE 30,10
1630 PRINT"Tschuess"
1640 LOCATE 1,23
1650 END
1700 REM Eingabe der Messwerte
1710 IF weiter=1 THEN RETURN
1720 anzahl=0
1730 CLS
1740 PRINT"Geben Sie die Messwerte ein.":PRINT:PRINT
"Beenden Sie die Messwerteingabe durch Eingabe von 3
33.33 als x-Wert"
1750 LOCATE 1,6
1760 anzahl=anzahl+1
1770 PRINT"x";anzahl;":";:INPUT x(anzahl)
1780 IF x(anzahl)=333.33 THEN 1830
1790 PRINT:PRINT"y";anzahl;":";:INPUT y(anzahl)
1800 LOCATE 1,6 : PRINT"

"
1810 LOCATE 1,8 : PRINT"

"
1820 GOTO 1750
1830 anzahl=anzahl-1
1840 LOCATE 1,15
1850 PRINT"Alle Eingaben i.o.     ( /n) ";:INPUT a$
1860 IF a$="" THEN 1880
```

```
1870 IF a$="n" THEN 1700 ELSE 1840
1880 RETURN
2000 REM Ausgabeunterprogramm
2010 CLS
2020 PRINT a1$
2030 PRINT"______________________________________________
_____________________________________"
2040 PRINT"a : ";a,,"b : ";b
2050 PRINT:PRINT"r^2         : ";r2u
2060 PRINT:PRINT"Abweichung : ";abw
2070            PRINT"             = ";USING"###.##";ab
wrel;:PRINT"%"
2080 LOCATE 1,23 : PRINT"Weiter mit beliebiger Taste
"
2090 IF INKEY$="" THEN 2090
2100 RETURN
2500 REM Berechnung der Guetekriterien
2510 REM Relative Quadratische Abweichungen
2520 abw=0
2530 FOR k=1 TO anzahl
2540 x=x(k)
2550 IF was=1 THEN yb = a + b*x
2560 IF was=2 THEN yb = a * EXP(b*x)
2570 IF was=3 THEN yb = a + b*LOG(x)
2580 IF was=4 THEN yb = a * x^b
2590 abw=abw + (yb-y(k))*(yb-y(k))
2600 NEXT k
2610 abw = SQR(abw/anzahl)
2620 REM Bestimmtheitsmass nach Ullmann
2630 sqay = 0 : sqmy = 0
2640 sum1=0
```

```
2650 FOR k=1 TO anzahl
2660 sum1=sum1+y(k)
2670 NEXT k
2680 mwy=sum1/anzahl
2690 FOR k=1 TO anzahl
2700 sqay = sqay + (y(k)-mwy)*(y(k)-mwy) : REM Quadr
atische Abweichungen der Messwerte
2710 x=x(k)
2720 IF was=1 THEN yb = a + b*x
2730 IF was=2 THEN yb = a * EXP(b*x)
2740 IF was=3 THEN yb = a + b*LOG(x)
2750 IF was=4 THEN yb = a * x^b
2760 sqmy = sqmy + (yb-mwy)*(yb-mwy)   : REM Quadrat
ische Abweichungen der berechneten y-Werte
2770 NEXT k
2780 r2u = sqmy/sqay
2790 REM relative Abweichung, bezogen auf den Mittel
wert
2800 abwrel = abw/mwy*100
2810 RETURN
```

10.3 Anpassung durch ein Polynom

Grundlagen

Wenn die Anpassung einer Meßwertreihe durch eine einfache Funktion nicht oder nur ungenügend gelingt, wird man als Kurventyp für die Anpassung ein Polynom wählen. Solch ein Polynom hat die allgemeine Funktionsgleichung

$$y = a_0 + a_1 x^1 + a_2 x^2 + \ldots + a_n x^n \tag{10.26}$$

n gibt hier den Grad des Polynoms an.

Die Ausgleichsforderung lautet wieder

$$[vv] = \sum_{i=0}^{N} (f(x_i) - y_i) = \text{Minimum!}$$

y_i ist der Meßwert und $f(x_i)$ ist der berechnete Funktionswert, der jeweils zu x_i gehört. Wenn z die Meßwertanzahl bezeichnet, läuft i im Gegensatz zu den beiden vorangegangenen Abschnitten nicht mehr von 1 bis z, sondern von 0 bis N, wobei $N = z - 1$ ist. Die Anzahl der Summationsglieder bleibt also gleich der Meßwertanzahl.

Ausführlich geschrieben lautet die Ausgleichsforderung:

$$[vv] = \sum_{i=0}^{N} (a_0 + a_1 x_i + \ldots + a_n x_i^n - y_i)^2 = \text{Minimum!} \qquad (10.27)$$

Gleichung (10.27) muß jetzt wie in Abschnitt 10.1 gezeigt partiell nach allen Variablen a_0 bis a_n abgeleitet werden; die so erhaltenen Gleichungen werden alle gleich 0 gesetzt, um die Bedingung für ein Minimum zu erfüllen.

$$\frac{d[vv]}{da_0} = 2\sum_{i=0}^{N} (a_0 + a_1 x_i + \ldots + a_1 x_i^n - y_i) \quad = 0$$

$$\frac{d[vv]}{da_0} = 2\sum_{i=0}^{N} (a_0 + a_1 x_i + \ldots + a_1 x_i^n - y_i) x_i = 0$$

$$\vdots \qquad\qquad\qquad\qquad \vdots$$

$$\frac{d[vv]}{da_0} = 2\sum_{i=0}^{N} (a_0 + a_1 x_i + \ldots + a_n x_i^n - y_i) x_i^n = 0$$

Dieses Gleichungssystem kann man auch folgendermaßen schreiben (alle Summen laufen hier von 0 bis N):

$$(N+1) a_0 + a_1 \sum x_i + \ldots + a_1 \sum x_i^n = \sum y_i$$

$$a_0 \sum x_i + a_1 \sum x_i^2 + \ldots + a_n \sum x_i^n = \sum y_i x_i$$

$$a_0 \sum x_i^n + a_1 \sum x_i^{n+1} + \ldots + a_n \sum x_i^{n+n} = \sum y_i x_i^n$$

Aus diesem Gleichungssystem kann man jetzt die Koeffizienten a_0 bis a_n berechnen und so das optimale Polynom aufstellen.

In Kurzform geschrieben lautet das Gleichungssystem:

$$\sum_{k=0}^{n} a_k \left(\sum_{i=0}^{N} x_i^k \cdot x_i^j \right) = \sum_{i=0}^{n} y_i \cdot x_i^j \qquad (j = 0,1,\ldots,n) \qquad (10.28)$$

Beispiel

Für die Meßwertreihe aus Abschnitt 10.1 soll das optimale Polynom 1. Grades bestimmt werden.

Meßwertanzahl = 6: $N = 5$
$n = 1$ (Grad des Polynoms)

Das Gleichungssystem lautet also (die Summen laufen von 0 bis 5):

$j = 0$: $\quad 6a_0 + a_1 \sum x_i = \sum y_i$

$j = 1$: $\quad a_0 \sum x_i + a_1 \sum x_i^2 = \sum y_i x_i$

Mit $\sum x_i = 18{,}5$; $\sum y_i = 8{,}6$; $\sum x_i^2 = 88{,}25$; $\sum y_i x_i = 44{,}25$ ergibt sich

$$6a_0 + 18{,}5 a_1 = 8{,}6$$
$$18{,}5 a_0 + 88{,}25 a_1 = 44{,}25$$

Hieraus errechnen sich a_0 und a_1:

$$a_0 = -0{,}318$$
$$a_1 = 0{,}568$$

Diese Werte stimmen mit den in Abschnitt 10.1 gefundenen überein; die Gütekriterien sind also

$$R^2 = 0{,}959; \quad dy = 0{,}270; \quad dyrel = 18{,}9\,\%$$

Wählt man als Ausgleichsfunktion ein Polynom höheren Grades, steigt die Anpassungsgenauigkeit; mit $n = 5$ erreicht man so z. B. ein Bestimmtheitsmaß von 1,0.

Literaturhinweise

G, W: [14], S. 80 ff; G, W: [18]; G, W: [22], S. 340 bis 343

Programmbeschreibung

Die Zeilennummern des Programms *Polanpass* beginnen erst bei 10000; auf diese Art und Weise soll erreicht werden, daß dieses Programm leichter mit dem zuvor beschriebenen Programm *Ausgleich* zusammengefaßt werden kann, so daß alle Ausgleichsprogramme dann in einem Programmpaket zusammengefaßt sind.

In
Zeile 10030 werden über das Unterprogramm in den
Zeilen 10200
bis 10310 die Ausgangswerte eingegeben. Entsprechend den Grundlagen wird dann in den
Zeilen 10500
bis 10800 das Gleichungssystem aufgestellt, das der Ausgleichsforderung genügt. Als Ergebnis hat man die Lösungsmatrix *lmat*. Die
Zeilen 10590
bis 10610
und
Zeilen 10630
bis 10650 berechnen lediglich x_i^k bzw. x_i^j; die etwas komplizierte Ausführung wurde gewählt, weil der CPC 464 wie schon erwähnt Schwierigkeiten beim Potenzieren negativer Zahlen hat; das Vorzeichen ist hier nicht immer richtig. In den
Zeilen 11000
bis 11200 werden die einzelnen Polynomglieder berechnet. Da es sich hierbei um die Lösung eines linearen Gleichungssystems handelt, wurde das entsprechende Programm aus Kapitel 8 übernommen. Anschließend werden in den
Zeilen 11400
bis 11590 wie in Abschnitt 10.2 die Gütekriterien berechnet. Die Ergebnisse werden über die
Zeilen 11800
bis 11950 ausgegeben. Zwischen den
Zeilen 12000
und 12090 sind die gleichen Weiterführungsmöglichkeiten wie im Programm *Ausgleich* gegeben.

Programmlänge: etwa 4,4 K.

Die Güte des verwendeten Lösungsalgorithmus kann man einfach dadurch überprüfen, daß man Werte einer beliebigen Funktion dem Programm als Meßwerte eingibt. Wählt man dann den Polynomgrad mindestens so hoch wie den tatsächlichen Polynomgrad, so findet das Programm normalerweise die richtigen Faktoren.

```
10000 REM Programm Polanpass 1.0
10010 DIM lmat (10,10),x(10),y(10),a1(10),a(10)
10020 MODE 2
10030 GOSUB 10200    : REM Eingabe der Wertetabelle

10040 GOSUB 10500   : REM Loesungsmatrixberechnung

10050 GOSUB 11000 : REM Berechnung der Polynomgliede
r
10060 GOSUB 11400 : REM Guetekriterien
10070 GOSUB 11800 : REM Ausgabe
10080 GOSUB 12000 : REM Weiterfuehrungsprogramm
10090 CLS : LOCATE 20,10 : PRINT"Na endlich. Das wir
d auch Zeit":PRINT:PRINT TAB(23);"Ich bin schon ganz
 muede."
10100 LOCATE 1,23
10110 END
10200 REM Eingabe der Wertetabelle
10210 CLS : IF weiter =1 THEN RETURN
10220 PRINT "Wieviele Wertepaare sollen eingegeben w
erden";:INPUT anz
10230 ERASE x : ERASE y  : DIM x(anz),y(anz)
10240  FOR k = 0 TO anz-1
10250 PRINT "x";k+1;"=";:  INPUT x(k)
10260 PRINT"y";k+1;"=";:INPUT y(k)
10270 NEXT k
10280 PRINT:PRINT"Eingabe ok ( /n)";:INPUT a$
10290 IF a$="n"THEN 10240
10300 IF a$<>""THEN 10280
10310 RETURN
10500 REM Loesungsmatrixberechnung
```

```
10510 CLS
10520 PRINT"Welchen Grad soll das Ausgleichspolynom
haben";:INPUT grad
10530 PRINT:PRINT:PRINT"Bitte warten Sie. Ich bin be
schaeftigt."
10540 ERASE lmat :DIM lmat(grad,grad+1)
10550 FOR j=0 TO grad
10560 FOR k=0 TO grad
10570 FOR i=0 TO anz-1
10580 IF x(i)<0 THEN 10590 ELSE 10610
10590 IF k/2=FIX(k/2) THEN 10610
10600 x1=-ABS(x(i)^k) : GOTO 10620
10610 x1= ABS(x(i)^k)
10620 IF x(i)<0 THEN 10630 ELSE 10650
10630 IF j/2=FIX(j/2) THEN 10650
10640 x2=-ABS(x(i)^j) : GOTO 10660
10650 x2= ABS(x(i)^j) :
10660 lmat(j,k) = lmat(j,k) + x1 * x2
10670 NEXT i
10680 NEXT k
10690 NEXT j
10700 REM b-Wert-Berechnung
10710 FOR j=0 TO grad
10720 FOR i=0 TO anz-1
10730 IF x(i)<0 THEN 10740 ELSE 10760
10740 IF j/2=FIX(j/2) THEN 10760
10750 x2=-ABS(x(i)^j) : GOTO 10770
10760 x2= ABS(x(i)^j)
10770 lmat(j,grad+1) = lmat(j,grad+1) + y(i)*x2
10780 NEXT i
10790 NEXT j
```

```
10800 RETURN
11000 REM Determinantenberechnung nach Gauss
11010 ERASE a : ERASE a1 : DIM a(grad,grad+1),a1(gra
d)
11020 FOR i=0 TO grad
11030 FOR k = 0 TO grad+1
11040 a(i,k)=lmat(i,k)
11050 NEXT k
11060 NEXT i
11070 FOR k = 0 TO grad
11080 FOR i = 0 TO grad
11090 IF i=k THEN GOTO 11150
11100 f = -a(i,k)/a(k,k)
11110 FOR l = 0 TO grad+1
11120 b= a(k,l)*f
11130 a(i,l)=a(i,l) + b
11140 NEXT l
11150 NEXT i
11160 NEXT k
11170 FOR i=0 TO grad
11180 a1(i) = a(i,grad+1) / a(i,i)
11190 NEXT i
11200 RETURN
11400 REM Guetekriterien
11410 mwy = 0 : sqmy = 0 : sqay = 0
11420 FOR k=0 TO anz-1
11430 mwy = mwy + y(k)
11440 NEXT k
11450 mwy = mwy/anz
11460 FOR k=0 TO anz-1
11470 sqay = sqay + (y(k)-mwy)*(y(k)-mwy)
```

```
11480 x=x(k) : GOSUB 11600 : REM Funktionswertberech
nung
11490 sqmy = sqmy + (yb-mwy)*(yb-mwy)
11500 NEXT k
11510 r2u = sqmy/sqay
11520 REM Quadratische Abweichungen
11530 FOR k=0 TO anz-1
11540 x=x(k) : GOSUB 11600 : REM Funktionswertberech
nung
11550 abw = abw + (y(k)-yb)*(y(k)-yb)
11560 NEXT k
11570 abw = SQR(abw/anz)
11580 abwrel = abw/mwy*100
11590 RETURN
11600 REM Funktionswertberechnung
11610 yb = 0
11620 FOR i=0 TO grad
11630 yb = yb + a1(i)*x^i
11640 NEXT i
11650 RETURN
11800 REM Ausgabe
11810 CLS
11820 PRINT"Anpassung durch ein Polynom"
11830 PRINT:PRINT"y = a0 + a1*x + a2*x^2 +...+an*x^n
"
11840 PRINT"------------------------------------------
--------------------------------------"
11850 PRINT
11860 FOR k=0 TO grad
11870 PRINT"a";k;": ";a1(k)
11880 NEXT k
```

```
11890 PRINT:PRINT:PRINT"r^2 : ";r2u
11900 PRINT:PRINT"Mittlere Abweichung : ";abw
11910   PRINT"                           = ";USING"###.##";
abwrel;:PRINT"%"
11920 LOCATE 1,23
11930 PRINT"Weiter mit beliebiger Taste"
11940 IF INKEY$="" THEN 11940
11950 RETURN
12000 REM Weiterfuehrungsprogramm
12010 CLS
12020 PRINT"Sollen die gleichen Messwerte fuer eine
weitere Berechnung verwendet werden
                         ( /n)
 ";:INPUT a$
12030 IF a$="" THEN weiter =1 : GOTO 10020
12040 IF a$<>"n" THEN 12000
12050 CLS
12060 PRINT"Soll eine weitere Rechnung mit anderen M
esswerten durchgefuehrt werden ( /n) ";:INPUT a$
12070 IF a$="" THEN RUN 10000
12080 IF a$<>"n" THEN 12050
12090 RETURN
```

Literaturverzeichnis

1. AYRES, F.: *Matrizen, Theorie und Anwendung*. Düsseldorf: McGraw-Hill, 1978.
2. BANDERMANN, F.: *Auswertung von Meßdaten*. In: Ullmanns Encyclopädie der technischen Chemie. Band 5: Analysen und Meßverfahren. Kelker, H. (Hrsg.). Weinheim: Verlag Chemie, ⁴1980.
3. BROECKER, H.C.: *Mathematik*. In: Ullmanns Encyclopädie der technischen Chemie. Band 1: Allgemeine Grundlagen der Verfahrens- und Reaktionstechnik. Bartholomé, E., u.a. (Hrsg). Weinheim: Verlag Chemie, ⁴1972
4. COURANT, R.: *Vorlesungen über Differential- und Integralrechnung*. Bd. 1: Funktionen einer Veränderlichen. Berlin: Springer, ³1955.
5. DUDEN: *Rechnen und Mathematik*. Mannheim: Bibliogr. Inst., ³1969.
6. FUESSEL, K.; JANSENN; SCHWERMANN, K.: *Mathematik für Fachoberschulen*. Köln-Porz: Stam, ²1975.
7. HAINZL, J.: *Mathematik für Naturwissenschaftler*. Stuttgart: Teubner, ²1977.
8. HARDTWIG, E.: *Fehler- und Ausgleichsrechnung*. B-I-Taschenbücher, Bd. 262/262a. Mannheim: Bibliogr. Inst., 1968.
9. KOWALSKY, H.J.: *Einführung in die lineare Algebra*. Berlin: de Gruyter, 1971.
10. KUESTER, F.W.; THIEL, A.; FISCHBECK, K.: *Logarithmische Rechentafeln*. Berlin: de Gruyter, ⁹⁴⁻⁹⁹1965.
11. LANGERS, F.: *Regressionsanalyse*. In: Ullmanns Encyclopädie der technischen Chemie. Bd. 4: Verfahrensentwicklung und Planung von Anlagen, Dokumentation. Bartholomé, E., u.a. (Hrsg). Weinheim: Verlag Chemie, ⁴1974
12. LAUTER, J.; KUYPERS, W.: *Mathematik für Gymnasien, Sekundarstufe I*. Bd. 4: Algebra I. Düsseldorf: Schwann, ⁶1981.

13. LENZ, H.: *Grundlagen der Elementarmathematik*. München, Wien: Hanser, [3]1976.
14. LUDWIG, R.: *Methoden der Fehler- und Ausgleichsrechnung*. Braunschweig: Vieweg, [1]1969.
15. MOESENTHIN, W.: *Einführung in die Mathematik*. Rinteln, München: Merkur, [3]1973.
16. NEISS, F.; LIERMANN, H.: *Determinanten und Matrizen*. Berlin, Heidelberg, New York: Springer, [8]1975.
17. SANDEN, H.v.: *Praktische Mathematik*. Stuttgart: Teubner, [5]1958.
18. SPAETH, H.: *Algorithmen für elementare Ausgleichsmodelle*. München, Wien: Oldenbourg, 1973
19. SPOEREL, I.: *Mathematik von der Schule zur Hochschule*. Berlin: de Gruyter, 1954.
20. STEIN, S.K.: *Einführungskurs höhere Mathematik*. Braunschweig, Wiesbaden: Vieweg, 1981.
21. STIEFEL, E.: *Einführung in die numerische Mathematik*. Stuttgart: Teubner, [5]1976.
22. STINGL, P.; ROTH, D.: *Mathematik für Fachhochschulen*. München, Wien: Hanser, [1]1977.
23. TASCHNER, R.: *Holzwege zur Mathematik I*. Bibliogr. Inst. (Hrsg). Mannheim, Wien, Zürich: B·I.-Wissenschaftsverlag, [1]1983.
24. TISCHEL, G.: *Grundkurs Analysis*. In: Einführung in die Mathematik für allgemeinbildende Schulen. Schröder, H.; Uchtmann, H. (Hrsg). Frankfurt am Main, Berlin, München: Diesterweg, [1]1974.
25. ZURMUEHL, R.: *Matrizen und ihre technischen Anwendungen*. Berlin, Göttingen, Heidelberg: Springer, [4]1964.

Stichwortverzeichnis

Mit diesem Buch entdecken Sie immer wieder neue, gute Seiten Ihres CPC 464 und überwinden den Frust vor mathematischen Problemen. Ob Schüler, Student, «fertiger» Techniker oder Ingenieur: Hier finden Sie die richtige Problemauswahl, verständliche und eindeutige Erläuterungen der Lösungswege, durchgerechnete Beispiele (ohne Benutzung der Programme) und – als Schwerpunkt – die Programme selbst. Damit können Sie natürlich auch unabhängig von den Grundlagenabschnitten arbeiten. Der Umgang mit den Programmen ist recht einfach, auch eventuelle Änderungen machen keine Schwierigkeiten dank der ausführlichen Beschreibungen.



ISBN 3-8023-0856-5

Ein Buch von CHIP, dem Mikrocomputer-Magazin

Harald Baumgart Höhere Mathematik auf dem CPC 464 CHIP WISSEN

AMSTRAD CPC

MÉMOIRE ÉCRITE
MEMORY ENGRAVED
MEMORIA ESCRITA



https://acpc.me/